ETL-0564

AD-A221 871

Parallel Algorithms for Computer Vision
Final Report

Tomaso Poggio

Massachusetts Institute of Technology
Artificial Intelligence Laboratory
545 Technology Square
Cambridge, Massachusetts 02139

April 1990

Prepared for:

Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209-2308

U.S. Army Corps of Engineers
Engineer Topographic Laboratories
Fort Belvoir, Virginia 22060-5546

E T L

US ARMY ENGINEER TOPOGRAPHIC LABORATORIES

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	16 April 1990	Final Annual 31 Aug 88 - 31 Jan 90

4. TITLE AND SUBTITLE

Parallel Algorithms for Computer Vision - Final Report

5. FUNDING NUMBERS

(C) DACA76-85-C-0010

6. AUTHOR(S)

Poggio, Tomaso

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Massachusetts Institute of Technology
Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DARPA
1400 Wilson Boulevard
Arlington, VA 22209-2308

US Army Engineer Topo Labs
Research Institute
Fort Belvoir, VA 22060-5546

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ETL-0564

11. SUPPLEMENTARY NOTES This subject was previously discussed in:

ETL-0456 Parallel Algorithms for Computer Vision Jan 1987 AD-A183 755
ETL-0495 " " " " ", Second Year Report Mar 1988 AD-A203 947
ETL-0529 " " " " ", Third Year Report Jan 1989 AD-A212 489

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release;
Distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The main effort in this project has been directed towards the development of an integrated vision system - the Vision Machine - based on a parallel supercomputer. The core of the Vision Machine is in fact a set of parallel algorithms for visual recognition and navigation in an unstructured environment. The present version of the Vision Machine has been demonstrated to process images in close to real time by (1) computing first several low-level cues, such as edges, stereo disparity, optical flow, color and texture, (2) integrating them to extract a cartoon-like description of the scene in terms of the physical discontinuities of surfaces, and (3) using this cartoon in a recognition stage, based on parallel model matching. In addition to the development of the parallel algorithms, their implementation and testing, we have also done substantial work in several areas that are very closely related. These include (1) design and fabrication of VLSI circuits to transfer to potentially cheap and fast hardware some of the software algorithms, (2) initial development of techniques to synthesize by learning vision algorithms, and (3) several projects involving autonomous navigation of small robots. (KP) ←

14. SUBJECT TERMS

Computer vision
Parallel algorithms
Connection Machine

15. NUMBER OF PAGES
62

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements.***

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

Preface

This report was prepared under Contract DACA76-85-C-0010 for the U.S. Army Engineer Topographic Laboratories, Fort Belvoir, Virginia 22060-5546 by Massachusetts Institute of Technology, Cambridge, Massachusetts. The Contracting Officer's Representative was George Lukes.

Contents

1 Overview

The main effort in the project has been directed towards the development of an integrated vision system - the Vision Machine - , based on a parallel supercomputer. The core of the Vision Machine is in fact a set of parallel algorithms for visual recognition and navigation in an unstructured environment. The present version of the Vision Machine has been demonstrated to process images in close to real time, by

1. computing first several low-level *cues*, such as edges, stereo disparity, optical flow, color and texture,
2. integrating them to extract a *cartoon-like* description of the scene in terms of the physical discontinuities of surfaces,
3. using this cartoon in a recognition stage, based on parallel model matching.

In addition to the development of the parallel algorithms, their implementation and testing, we have also done substantial work in several areas that are very closely related:

- design and fabrication of VLSI circuits – analog and digital – to transfer to potentially cheap and very fast hardware some of the software algorithms of the Vision Machine,
- initial development of techniques to synthesize by learning vision algorithms or improve them with the use of pertinent examples,
- several projects involving autonomous navigation of small robots, recognition techniques and computation of salient contours.

In the following we will provide background information on all of these items. Additional details can be found in the references cited.

2 The Vision Machine

2.1 Introduction: The Vision Machine Project

Computer vision has developed algorithms for several early vision processes, such as edge detection, stereopsis, motion, texture, and color, which give separate cues as to the distance from the viewer of three-dimensional surfaces, their shape, and their material properties. Biological vision systems, however, greatly outperform computer vision programs. It is clear that one of the keys to the reliability, flexibility, and robustness of biological vision systems in unconstrained environments is their ability to integrate many different visual cues. For this reason, we have developed and continue to develope a *Vision Machine* system to explore the issue of integration of early vision modules. The system also serves the purpose of developing parallel vision algorithms, since its main computational engine is a parallel supercomputer, the Connection Machine.

The idea behind the Vision Machine is that the main goal of the integration stage is to compute a map of the visible discontinuities in the scene, somewhat similar to a cartoon or a line-drawing. There are several reasons for this. First, experience with existing model-based recognition algorithms suggest that the critical problem in this type of recognition is to obtain a reasonably good map of the scene in terms of features such as edges and corners. The map does not need to be perfect (human recognition works with noisy and occluded line drawings) and, of course, it cannot be. But it should be significantly cleaner than the typical map provided by an edge detector. Second, discontinuities of surface properties are the most important locations in a scene. Third, we have argued that discontinuities are ideal for integrating information from different visual cues.

It is also clear that there are several different approaches to the problem of how to integrate visual cues. Let us list some of the obvious possibilities:

1) There is no active integration of visual processes. Their individual outputs are "integrated" at the stage at which they are used, for example by a navigation system. This is the approach advocated by Brooks (1987). While it makes sense for automatic, insect-like, visuo-motor tasks such as tracking a target or avoiding obstacles (e.g., the fly's visuo-motor system (Poggio and Reichardt, 1976)), it seems quite unlikely for visual perception in the wide sense.

2) The visual modules are so tightly coupled that it is impossible to consider visual modules as separate, even in a first order approximation. This view is unattractive on epistemological, engineering and psychophysical grounds.

3) The visual modules are coupled to each other and to the image data in a parallel fashion – each process represented as an array coupled to the arrays associated with the other processes. This point of view is in the tradition of Marr's $2\frac{1}{2}$-D sketch, and especially of the "intrinsic images" of Barrow and Tenenbaum (1978). Our present scheme is of this type, and exploits the machinery of

Markov Random Field (MRF) models.

4) Integration of different vision modalities is taking place in a task-dependent way at specific locations - not over the whole image - and when it is needed - therefore not at all times. This approach is suggested by psychophysical data on visual attention and by the idea of visual routines (Ullman, 1984; see also Hurlbert and Poggio, 1986; Mahoney, 1986; Buelthoff and Mallot, 1987).

We have actively explored, in the framework of the contract Parallel Vision Algorithms, the third of these approaches. We believe that the last two approaches are compatible with each other. In particular, visual routines may operate on maps of discontinuities such as those delivered by the present Vision Machine, and therefore be located after a parallel, automatic integration stage. In real life, of course, it may be more a matter of coexistence. We believe, in fact, that a control structure based on specific knowledge about the properties of the various modules, the specific scene and the specific task will be needed in a later version of the Vision Machine to overview and control the MRF integration stage itself and its parameters. It is possible that the integration stage should be much more goal-directed that what our present methods (MRF based) allow. The main goal of our work is to find out whether this is true.

The Vision Machine project had a number of goals. It provided a focus for developing parallel vision algorithms and for studying how to organize a real-time vision system on a massively parallel supercomputer. It attempts to alter the usual paradigm of computer vision research over the past years: choose a specific problem, for example stereo, find an algorithm, and test it in isolation. The Vision Machine has allowed us to develop and test an algorithm in the context of the other modules and the requirements of the overall visual task, above all visual recognition. For this reason, the project was more than an experiment in integration and parallel processing: it was and still is a laboratory for our theories and algorithms.

Finally, the ultimate goal of the Vision Machine project is no less than the ultimate goal of vision research: to build a vision system that achieves human-level performance.

2.2 The Vision Machine System

The overall organization of the system is shown in Figure 1. The image(s) are processed in parallel through independent algorithms or modules corresponding to different visual cues. Edges are extracted using Canny's edge detector. The stereo module computes disparity from the left and right images. The motion module estimates an approximation of the optical flow from pairs of images in a time sequence. The texture module computes texture attributes (such as density and orientation of textons (see Voorhees, 1987)). The color algorithm provides an estimate of the spectral albedo of the surfaces, independently of the *effective illumination*, that is, illumination gradients and shading effects, as suggested by Hurlbert and Poggio (see Hurlbert and Poggio, 1985).

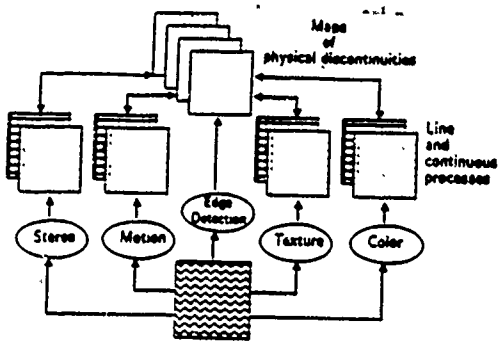

Figure 1: *Overall organization of the Vision Machine.*

The measurements provided by the early vision modules are typically noisy, and possibly sparse (for stereo and motion). They are smoothed and made dense by exploiting known constraints within each process (for instance, that disparity is smooth). This is the stage of *approximation* and *restoration* of data, performed using a Markov Random Field model. Simultaneously, discontinuities are found in each cue. Prior knowledge of the behavior of discontinuities is exploited, for instance, the fact that they are continuous lines, not isolated points. Detection of discontinuities is aided by the information provided by brightness edges. Thus each cue, disparity, optical flow, texture, and color, is coupled to the edges in brightness.

The full scheme involves finding the various types of physical discontinuities in the surfaces, *depth discontinuities* (extremal edges and blades), *orientation discontinuities, specular edges, albedo edges* (or marks), and *shadow edges*, and coupling them with each other and back to the discontinuities in the visual cues (as illustrated in Figure 1 and suggested by Geiger and Weinshall, 1988 and Gamble, Geiger, Poggio and Weinshall, 1989). So far we have implemented only the coupling of brightness edges to each of the cues provided by the early algorithm. As we will discuss later, the technique we use to approximate, to simultaneously detect discontinuities, and to couple the different processes, is based on MRF models. The output of the system is a set of labeled discontinuities of the surfaces around the viewer. Thus the scheme – an instance of inverse optics – computes *surface properties*, that is attributes of the physical world and not anymore of the images. Notice that we attempt to find discontinuities in surface properties and therefore qualitative surface properties: the *inverse optics* paradigm does **not** imply that physical properties of the surfaces, such as depth or reflectance, should be extracted ***precisely, everywhere.*** These discontinuities, taken together, represent a "cartoon" of the original scene which can be used for recognition and navigation (along with, if needed, interpolated depth, motion, texture and color fields). As yet we did not integrate our ongoing work on grouping in the Vision Machine. We expect to use a saliency operation on the output of the edge detection process possibly before the use of intensity edges by the MRF stage. The grouping based on T-junctions (Beymer, in preparation) should take place on the intensity edges at the same level as the MRF stage. Initial work in recognition has been integrated in the system. The Vision Machine has been demonstrated working form images to recognition through the integration of visual cues.

The plan of this section is as follows. We will first review the current hardware of the Vision Machine: the eye-head system and the Connection Machine. We will then describe in some detail

each of the early vision algorithms that are presently running and are part of the system. After this, the integration stage will be discussed. We will analyze some results, and illustrate the merits and the pitfalls of our present system. The last chapter will discuss a real-time visual system, and some ideas on how to put the system into VLSI circuits of analog and digital type.

2.3 Hardware

2.3.1 The Eye-Head System

Because of the scope of the Vision Machine project, a general purpose image input device is required. Such a device is the eye-head system. Here we discuss its current and future configurations.

The eye-head system consists of two CCD cameras, which act as eyes, mounted on a variable-attitude platform, which acts as the head. Inspired by biology, the apparatus is configured such that the head moves the eyes as a unit, while allowing the eyes to point independently. Each eye is equipped with a motorized zoom lens ($F1.4$, focal length from 12.5 to 75mm), allowing control of the iris, focus, and focal length by the host computer (currently a Symbolics 3600 Lisp Machine). Other hardware allows for repeatable calibration of the entire apparatus.

Because of the size and weight of the motorized lenses, it would be impractical to achieve eye movement by pointing the camera/lens assemblies directly. Instead, each assembly is mounted rigidly on the head, with eye movement achieved indirectly. In front of each lens is a pair of front surface mirrors, each of which can be pivoted by a galvanometer, providing two degrees of freedom in aiming the cameras. At the expense of a more complicated imaging geometry, we get a simple and fast pointing system for the eyes.

The head is attached to its mount via a spherical joint, allowing head rotation about two orthogonal axes (pan and tilt). Each axis is driven by a stepper motor coupled to its drive shaft through a harmonic drive. The latter provides a large gear ratio in conjunction with very little mechanical backlash. Under control of the stepper motors, the head can be panned 180 degrees from left to right, and tilted 90 degrees (from vertical-down to horizontal). Each of the stepper motors is provided with an optical shaft encoder for shaft position feedback (a closed-loop control scheme is employed for the stepper motors). The shaft encoders also provide an index pulse (one per revolution) which is used for joint calibration in conjunction with mechanical limit switches. The latter also protect the head from damage due to excessive travel.

The overall control system for the eye-head system is distributed over a micro-processor network (UNET) developed at the MIT AI Laboratory for the control of vision/robotics hardware. The UNET is a "multi-drop" network supporting up to 32 micros, under the control of a single host. The

micros normally function as network slaves, with the host acting as the master. In this mode the micros only "speak when spoken to," responding to various network operations either by receiving information (command or otherwise) or by transmitting information (such as status or results). Associated with each micro on the UNET is a local 16-bit bus (UBUS), which is totally under the control of the micro. Peripheral devices such as motor drivers, galvanometer drivers, and pulse width modulators (PWMs), to name a few, can be interfaced at this level.

At present, three micro-processors are installed on the eye-head UNET: one each for the galvanometers, motorized lenses, and stepper motors. The processors currently employed are based on the Intel 8051. Each of these micros has an assortment of UBUS peripherals under its control. By making these peripherals sufficiently powerful, each micro's control task can remain simple and manageable. Code for the micros, written in both assembly language and C, is facilitated by a Lisp-based debugging environment.

A single major enhancement remains for the eye-head system. Currently, a Symbolics Lisp Machine acts as the host processor for the UNET. In the fall of '89, an intermediate real-time processor will be placed between the Lisp Machine and the UNET, acting as master of the latter. The real-time processor (referred to as the DSP, being based on a Digital Signal Processor chip) will relieve the Lisp Machine of all the UNET protocol tasks, as well as various low-level, real-time control tasks for which the Lisp Machine is ill-suited. Among the tasks envisioned for the DSP is optimal position estimation of moving targets.

2.3.2 Our Computational Engine: The Connection Machine

The Connection Machine is a powerful fine-grained parallel machine which has proven useful for implementation of vision algorithms. In implementing these algorithms, several different models of using the Connection Machine have emerged, since the machine provides several different communication modes. The Connection Machine implementation of algorithms can take advantage of the underlying architecture of the machine in novel ways. We describe here several common, elementary operations which recur throughout the following discussion of parallel algorithms.

The Connection Machine

The CM-2 version of the Connection Machine (Hillis, 1985) is a parallel computing machine with between 16K and 64K processors, operating under a single instruction stream broadcast to all processors. It is a Single Instruction Multiple Data (SIMD) machine; all processors execute the same control stream. Each processor is a simple 1-bit processor, currently with 64K bits of memory, optionally with a floating point arithmetic accelerator, shared among 16 (or 32) processors. There are two modes of communication among the processors: the NEWS network and the *router*. The

NEWS network (so-called because the connections are in the four cardinal directions) provides rapid direct communication between neighboring processors in an rectangular grid of arbitrary dimension. For example, 64K processors could be configured into a two-dimensional 256×256 grid, or into a four-dimesional $64 \times 64 \times 4 \times 4$ grid. The second mode of communication is the *router*, which allows messages to be sent from any processor to any other processor in the machine. The processors in the Connection Machine can be envisioned as being the vertices of a 16-dimensional hypercube (in fact, it is a 12-dimensional hypercube; at each vertex of the hypercube resides a chip containing 16 processors). Each processor in the Connection Machine is identified by its hypercube address in the range $0 \ldots 65535$, imposing a linear order on the processors. This address denotes the destination of messages handled by the router. Messages pass along the edges of the hypercube from source processors to destination processors. The Connection Machine also has facilities for returning to the host machine the result of various operations on a field in all processors; it can return the global maximum, minimum, sum, logical AND, and logical OR of the field.

The floating-point arithmetic accelerator, which may optionally be added to the Connection Machine, provides a significant increase in the speed of both single and double precision computations. One floating-point processor chip serves a pair Connection Machine processor chips with 32 total processors in a pipelined fashion, and can produce a speed-up of more than a factor of twenty.

To allow the machine to manipulate data structures with more than 64K elements, the Connection Machine supports *virtual processors*. A single physical processor can operate as a set of multiple virtual processors by serializing operations in time, and by partitioning the memory of each processor. This is otherwise invisible to the user. Connection Machine programs utilize Common Lisp syntax, in a language called *Lisp, and are manipulated in the same fashion as Lisp programs.

Powerful Primitive Operations

Many vision problems must be solved by a combination of communication modes on the Connection Machine. The design of these algorithms takes advantage of the underlying architecture of the machine in novel ways. There are several common, elementary operations used in this discussion of parallel algorithms: routing operations, scanning, and distance doubling.

Routing

Memory in the Connection Machine is associated with processors. Local memory can be accessed rapidly. Memory of processors nearby in the NEWS network can be accessed by passing it through the processors on the path between the source and the destination. At present, NEWS accesses in the machine are made in the same direction for all processors. The *router* on the Connection Machine provides parallel reads and writes among processor memory at arbitrary distances and with arbitrary patterns. It uses a packet-switched message routing scheme to direct messages along the hypercube connections to their destinations. This powerful communication mode can be used to reconfigure completely, in one parallel write operation taking one router cycle, a field of information in the machine. The Connection Machine supplies instructions so that many processors can read

from the same location or write to the same location, but since these memory references can cause significant delay, we will usually only consider exclusive read, exclusive write instructions. We will usually not allow more than one processor to access the memory of another processor at one time. The Connection Machine can combine messages at a destination by various operations, such as logical AND, inclusive OR, summation, and maximum or minimum.

Scanning

The *scan* operations (Blelloch, 1987) can be used to simplify and speed up many algorithms. They directly take advantage of the hypercube connections underlying the router, and can be used to distribute values among the processors and to aggregate values using associative operators. Formally, the *scan* operation takes a binary associative operator $\oplus$, with identity 0, and an ordered set $[a_0, a_1, \ldots, a_{n-1}]$, and returns the set $[a_0, (a_0 \oplus a_1), \ldots, (a_0 \oplus a_1 \oplus \ldots \oplus a_{n-1})]$. This operation is sometimes referred to as the *data independent prefix operation.* Binary associative operators include *minimum, maximum,* and *plus.*

The four scan operations *plus-scan, max-scan, min-scan,* and *copy-scan* are implemented in microcode, and take about the same amount of time as a routing cycle. The *copy-scan* operation takes a value at the first processor and distributes it to the other processors. These scan operations can take *segment bits* that divide the processor ordering into segments. The beginning of each segment is marked by a processor whose segment bit is set, and the scan operations start over again at the beginning of each segment.

The *scan* operations also work using the NEWS addressing scheme, termed *grid-scans.* These compute the sum, and quickly find the maximum, copy, or number values along rows or columns of the NEWS grid.

For example, *grid-scans* can be used to find, for each pixel, the sum of a square region with width $2m + 1$ centered at the pixel. This sum is computed using the following steps. First, a *plus-scan* operation accumulates partial sums for all pixels along the rows. Each pixel then gets the result of the scan from the processor m in front of it and m behind it; the difference of these two values represents the sum, for each pixel, of its neighborhood along the row. We now execute the same calculation on the columns, resulting in the sum, for each pixel, of the elements in its square. The whole process only requires a few *scans* and routing operations, and runs in time independent of the size of m. The summation operations are generally useful to accumulate local support in many of our algorithms, such as stereo and motion.

Distance Doubling

Another important primitive operation is *distance doubling* (Wyllie, 1979; Lim, 1986), which can be used to compute the effect of any binary, associative operation, as in *scan,* on processors linked in a list or a ring. For example, using *max, distance doubling* can find the extremum of a field

contained in the processors. Using message-passing on the router, *distance doubling* can propagate the extreme value to all processors in the ring of N processors in $O(\log N)$ steps. Each step involves two *send* operations. Typically, the value to be maximized is chosen to be the hypercube address. At termination, each processor in the ring knows the label of the maximum processor in the ring, hereafter termed the *principal processor*. This labels all connected processors uniquely, and nominates a processor as the representative for the entire set of connected processors. At the same time, the distance from the *principal* can be computed in each processor. Each processor initially, at step 0, has the address of the next processor in the ring, and a value which is to be maximized. At the termination of the i^{th} step, a processor knows the addresses of processors $2^i + 1$ away, and the maximum of all values within 2^{i-1} processors away. In the example, the maximum value has been propagated to all 8 processors in $\log 8 = 3$ steps.

2.4 Early Vision Algorithms and their Parallel Implementation

2.4.1 Edge Detection

Edge detection is a key first step in correctly identifying physical changes. The apparently simple problem of measuring sharp brightness changes in the image has proven to be difficult. It is now clear that edge detection should be intended not simply as finding "edges" in the images, an ill-defined concept in general, but as measuring appropriate derivatives of the brightness data. This involves the task-dependent use of different two-dimensional derivatives. In many cases, it is appropriate to mark locations corresponding to appropriate critical points of the derivative such as maxima or zeroes. In some cases, later algorithms based on these binary features (presence or absence of edges) may be equivalent, or very similar, to algorithms that directly use the continuous value of the derivatives. A case in point is provided by our stereo and motion algorithms, to be described later. As a consequence, one should not always make a sharp distinction between edge-based and intensity based algorithms; the distinction is more blurred, and in some cases it is almost a matter of implementation.

In our current implementation of the Vision Machine, we are using two different kinds of edges. The first consists of zero-crossings in the Laplacian of the image filtered through an appropriate Gaussian. The second consists of the edges found by Canny's edge detector. Zero-crossings can be used by our stereo and motion algorithms (though we have mainly used Canny's edges at fine resolution). Canny's edges (at a coarser resolution) are input to the MRF integration scheme.

Zero-Crossings

Because the derivative operation is ill-posed, we need to filter the resultant data through an appropriate low-pass filter (Torre and Poggio, 1986). The filter of choice (but not the only possibility!) is a Gaussian at a suitable spatial scale. An interesting and simple implementation of Gaussian convolution relies on the binomial approximation to the Gaussian distribution. This algorithm requires only integer addition, shifting, and local communication on the 2D mesh, so it can be implemented on a simple 2D mesh architecture (such as the NEWS network on the Connection Machine).

The Laplacian of a Gaussian is often approximated by the difference of Gaussians. The Laplacian of a Gaussian can also be computed by convolution with a Gaussian followed by convolution with a discrete Laplacian; we have implemented both on the Connection Machine. To detect zero-crossings, the computation at each pixel need only examine the sign bits of neighboring pixels.

Canny Edge Detection

The Canny edge detector is often used in image understanding. It is based on directional derivatives, so it has improved localization. The Canny edge detector on the Connection Machine consists of the following steps:

- Gaussian smoothing,
- Directional derivative,
- Non-maximum suppression,
- Thresholding with hysteresis.

Gaussian filtering, as described above, is a local operation. Computing directional derivatives is also local, using a finite difference approximation referencing only local neighbors in the image grid.

Non-maximum Suppression

Non-maximum suppression selects as edge candidates those pixels for which the gradient magnitude is maximal in the direction of the gradient. This involves interpolating the gradient magnitude between each of two pairs of adjacent pixels among the eight neighbors of a pixel, one forward in the gradient direction, and one backward. However, it may not be crucial to use interpolation, in which case magnitudes of neighboring values can be directly compared.

Thresholding with Hysteresis

Thresholding with hysteresis eliminates weak edges due to noise, using the threshold, while connecting extended curves over small gaps using hysteresis. Two thresholds are computed, *low* and *high*, based on an estimate of the noise in the image brightness. The non-maximum suppression step selects those pixels where the gradient magnitude is maximal in the direction of the gradient. In the thresholding step, all selected pixels with gradient magnitude below *low* are eliminated. All

pixels with values above *high* are considered as edges. All pixels with values between *low* and *high* are edges if they can be connected to a pixel above *high* through a chain of pixels above *low*. All others are eliminated.

This is a spreading activation operation; it propagates information along a set of connected edge pixels. The algorithm iterates, in each step marking as *edge* pixels any *low* pixels adjacent to *edge* pixels. When no pixels change state, the iteration terminates, taking $O(m)$ steps, a number proportional to the length m of the longest chain of *low* pixels which eventually become *edge* pixels. The running time of this operation can be reduced to $O(\log m)$, using *distance doubling*.

Noise Estimation

Estimating noise in the image can be done by analyzing a histogram of the gradient magnitudes. Most computational implementations of this step perform a global analysis of the gradient magnitude distribution, which is essentially non-local; we have had success with a Connection Machine implementation using local histograms. The thresholds used in Canny edge detection depend on the final task for which the edges are used. A conservative strategy can use an arbitrary low threshold to eliminate the need for the costly processing required to accumulate a histogram. Where a more precise estimate of noise is needed, it may be possible to find a scheme that uses a coarse estimate of the gradient magnitude distribution, with minimal global communication.

2.4.2 Stereo

The Drumheller-Poggio parallel stereo algorithm (Drumheller and Poggio, 1986) runs as part of the Vision Machine. Disparity data produced by the algorithm comprise one of the inputs to the MRF-based integration stage of the Vision Machine. We are exploring various extensions of the algorithm, as well as the possible use of feedback from the integration stage. In this section, we will review the algorithm briefly, then proceed to a discussion of current research.

The stereo algorithm runs on the Connection Machine system with good results on natural scenes in times that are typically on the order of one second. The stereo algorithm is presently being extended in the context of the Vision Machine project.

The Drumheller-Poggio Stereo Algorithm

Stereo matching is an ill-posed problem (see Bertero et al., 1988) that cannot be solved without taking advantage of natural constraints. The *continuity constraint* (see, for instance, Marr and Poggio, 1976) asserts that the world consists primarily of piecewise smooth surfaces. If the scene contains no transparent objects, then the *uniqueness constraint* applies: there can be only one match along the left or right lines of sight. If there are no narrow occluding objects, the *ordering*

constraint (Yuille and Poggio, 1984) holds: any two points must be imaged in the same relative order in the left and right eyes.

The specific *a priori* assumption on which the algorithm is based is that the disparity, that is, the depth of the surface, is locally constant in a small region surrounding a pixel. It is a restrictive assumption which, however, may be a satisfactory *local* approximation in many cases (it can be extended to more general surface assumptions in a straightforward way, but at a high computational cost). Let $E_L(x,y)$ and $E_R(x,y)$ represent the left and the right image of a stereo pair, or some transformation of it, such as filtered images or a map of the zero-crossings in the two images (more generally, they can be maps containing a feature vector at each location (x,y) in the image).

We look for a discrete disparity $d(x,y)$ at each location x,y in the image that minimizes

$$\|E_L(x,y) - E_R(x + d(x,y), y)\|_{\text{patch}_i}$$

where the norm is a summation over a local neighborhood centered at each location (x,y); $d(x)$ is assumed constant in the neighborhood. The previous equation implies that we should look at each (x,y) for $d(x,y)$ such that

$$\int_{\text{patch}_i} (E_L(x,y)E_R(x + d(x,y), y))^2 dxdy \qquad (1)$$

is maximized.

The algorithm that we have implemented on the Connection Machine is actually somewhat more complicated, since it involves geometric constraints that affect the way the maximum operation is performed (see Drumheller and Poggio, 1986). The implementation currently used in the Vision Machine at the AI Laboratory uses the maps of Canny edges obtained from each image for E_L and E_R.

In more detail, the algorithm is composed of the following steps:

1) Compute features for matching.

2) Compute potential matches between features.

3) Determine the degree of continuity around each potential match.

4) Choose correct matches based on the constraints of continuity, uniqueness, and ordering.

Potential matches between features are computed in the following way. Assuming that the images are registered so that the epipolar lines are horizontal, the stereo matching problem becomes one-dimensional: an edge in the left image can match any of the edges in the corresponding horizontal scan line in the right image. Sliding the right image over the left image horizontally, we compute a set of *potential match planes*, one for each horizontal disparity. Let $p(x,y,d)$ denote the value of the (x,y) entry of the potential match plane at disparity d. We set $p(x,y,d) = 1$ if there is an edge

at location (x, y) in the left image and a compatible edge at location $(x - d, y)$ in the right image; otherwise, set $p(x, y, d) = 0$. In the case of the DOG edge detector, two edges are compatible if the sign of the convolution for each edge is the same.

To determine the degree of continuity around each potential match (x, y, d), we compute a local support score $s(x, y, d) = \sum_{Patch} p(x, y, d)$, where *patch* is a small neighborhood of (x, y, d) within the *dth* potential match plane. In effect, nearby points in *patch* can "vote" for the disparity d. The score $s(x, y, d)$ will be high if the continuity constraint is satisfied near (x, y, d), i.e., if *patch* contains many votes. This step corresponds to the integral over the patch in the last equation.

Finally, we attempt to select the correct matches by applying the uniqueness and ordering constraints (see above). To apply the uniqueness constraint, each match suppresses all other matches along the left and right lines of sight with weaker scores. To enforce the ordering constraint, if two matches are not imaged in the same relative order in left and right views we discard the match with the smaller support score. In effect, each match suppresses matches with lower scores in its forbidden zone (Yuille and Poggio, 1984). This step corresponds to choosing the disparity value that maximizes the integral of the last equation.

Improvements

Using this algorithm as a base, we have explored several of the following topics:

Detection of Depth Discontinuities

The Marr-Poggio continuity constraint is both a strength and a weakness of the stereo algorithm. Favoring continuous disparity surfaces reduces the solution space tremendously, but also tends to smooth over depth discontinuities present in the scene. Consider what happens near a linear depth discontinuity, say a point near the edge of a table viewed from above. The square local support neighborhood for the point will be divided between points on the table and points on the floor; thus, almost half of the votes will be for the wrong disparity.

One solution to this problem is feedback from the MRF integration stage. We can take the depth discontinuities located by the integration stage (using the results from a first pass of the stereo algorithm, among other inputs) and use them to restrict the local support neighborhoods so that they do not span discontinuities. In the example mentioned above, the support neighborhood would be trimmed to avoid crossing the discontinuity between the table and the floor, and thus would not pick up spurious votes from the floor.

We can also try to locate discontinuities by examining intermediate results of the stereo algorithm. Consider a histogram of votes vs. disparity for the table/floor example. For a support region centered near the edge of the table, we expect to see two strong peaks: one at the disparity of the floor, and the other at the disparity of the table. Therefore a bimodal histogram is strong evidence for the presence of a discontinuity.

These two ideas can be used in conjunction. Discontinuity detection within stereo can take advantage of the extra information provided by the vote histograms. By passing better depth data (and perhaps candidate discontinuity locations) to the integration stage, we improve the detection of discontinuities at the higher level.

Improving the Stereo Matcher

The original Drumheller-Poggio algorithm matched DOG zero-crossings, where the local support score counted the number of zero-crossings in the left image patch matching edges in the right image patch at a given disparity. We have modified the matcher in a variety of ways.

1) Canny edges. The matcher now uses edges derived by a parallel implementation of the Canny edge detector (Canny, 1983; Little et al., 1987) rather than DOG zero-crossings, for better localization.

2) Gradient direction constraint. We allow two Canny edges to match only if the associated brightness gradient directions are aligned within a parameterized tolerance. This is analogous to the restriction in the Marr-Poggio-Grimson stereo algorithm (Grimson, 1981), where two zero-crossings can match only if the directions of the DOG gradients are approximately equal. Matching gradient orientations is a tighter constraint than matching the sign of the DOG convolution. Furthermore, the DOG sign is numerically unstable for horizontally oriented edges.

3) The scores are now normalized to take into account the number of edges in the left and right image patches eligible to match, so that patches with high edge densities do not generate artificially high scores. We plan to change the matcher so that edges that fail to match would count as negative evidence by reducing the support score, but this has not yet been implemented.

In the near future, we will explore matching brightness values as well as edges, using a cross-correlation approach similar to that of Little, Buelthoff and Poggio (1987) for motion estimation.

Identifying Areas that are Outside of the Matcher's Disparity Range

The stereo algorithm searches a limited disparity range, selected manually. Every potential match in the scene (an edge with a matching edge at some disparity) is assigned the in-range disparity with the highest score, even though the correct disparity may be out of range. How can we tell when an area of the scene is out of range? The most effective approach that we have attempted to date is to look for regions with low matching scores. Two patches that are incorrectly matched will, in general, produce a low matching score.

Memory-Based Registration and Calibration

Registration of the image pair for the stereo algorithm is done by presenting to the system a pattern of dots, roughly on a sparse grid, at the distance around which stereo has to operate. The registration is accomplished using a warping computed by matching the dots from the left and right images. The dots are sparse enough that matching is unambiguous. The matching defines a

warping vector for each dot; at other points the warping is computed by bilinear interpolation of the two components of warping vectors. The warping necessary for mapping the right image onto the left image is then stored. Prior to stereo-matching, the right image is warped according to the pre-stored addresses by sending each pixel in the right image to the processor specified in the table. The warping table corrects for deformations, including those due to vertical disparities and rotations, those due to the image geometry (errors in the alignment of the cameras, perspective projection, errors introduced by the optics, etc.). We plan to store several warping tables for each of a few convergence angles of the two cameras (assuming symmetric convergence). We conjecture that simple interpolation can yield sufficiently accurate warping tables for fixation angles intermediate to the ones stored. Notice that these tables are independent of the position of the head. Absolute depth is not the concern here (we are not using it in our present Vision Machine), but it could easily be recovered from knowledge of the convergence angle. Notice also that the whole registration scheme has the flavor of a learning process. Convergence angles are inputs and warping tables are the outputs of the modules; the set of angles, together with the associated warping tables, represent the set of input-output examples. The system can "generalize" by interpolating between warping tables and providing the warping corresponding to a vergence angle that does not appear in the set of "examples". Calibration of disparity to depth could be done in a similar way.

2.4.3 Motion

The motion algorithm computes the optical flow field, a vector field that approximates the projected motion field. The procedure produces sparse or dense output, depending on whether it uses edge features or intensities. The algorithm assumes that image displacements are small, within a range $(\pm\delta, \pm\delta)$. It is also assumed that the optical flow is locally constant in a small region surrounding a point. This assumption is strictly only true for translational motion of 3D planar surface patches parallel to the image plane. It is a restrictive assumption which, however, may be a satisfactory *local* approximation in many cases. Let $E_t(x,y)$ and $E_{t+\Delta t}(x,y)$ represent transformations of two discrete images separated by time interval Δt, such as filtered images, or a map of the brightness changes in the two images (more generally, they can be maps containing a feature vector at each location (x,y) in the image) (Kass, 1986; Nishihara, 1984).

We look for a discrete motion displacement $\underline{v} = (v_x, v_y)$ at each location x, y in the image that minimizes

$$\|E_t(x,y) - E_{t+\Delta t}(x + v_x\Delta t, y + v_y\Delta t)\|_{\text{patch}_i} = \min)$$

where the norm is a summation over a local neighborhood centered at each location (x,y); $\underline{v}(x,y)$ is assumed constant in the neighborhood. The previous equation implies that we should look at

each (x, y) for $\underline{v} = (v_x, v_y)$ such that

$$\int_{\text{patch}_i} (E_t(x,y) - E_{t+\Delta t}(x + v_x\Delta t, y + v_y\Delta t))^2 dxdy \tag{2}$$

is minimized. Alternatively, one can maximize the negative of the integrated result. The last equation represents the sum of the pointwise squared differences between a patch in the first image centered around the location (x, y) and a patch in the second image centered around the location $(x + v_x\Delta t, y + v_y\Delta t)$.

This algorithm can be translated easily into the following description. Consider a network of processors representing the result of the integrand in the previous expression. Assume for simplicity that this result is either 0 or 1 (this is the case if E_t and $E_{t+\Delta t}$ are binary feature maps). The processors hold the result of differencing (taking the logical "exclusive or") the right and left image map for different values of (x, y) and v_x, v_y. The next stage, corresponding exactly to the integral operation over the patch, is for each processor to summate the total in an (x, y) neighborhood at the same disparity. Note that this summation operation is efficiently implemented in the Connection Machine using *scan* computations. Each processor thus collects a vote indicating support that a patch of surface exists at that displacement. The algorithm iterates over all displacements in the range $(\pm\delta, \pm\delta)$, recording the values of the integral for each displacement. The last stage is to choose $\underline{v}(x, y)$ among the displacements in the allowed range that maximizes the integral. This is done by an operation of "non-maximum suppression" across velocities out of the finite allowed set. at the given (x, y), the processor is found that has the maximum vote. The corresponding $\underline{v}(x, y)$ is the velocity of the surface patch found by the algorithm. The actual implementation of this scheme can be simplified so that the "non-maximum suppression" occurs during iteration over displacements, so that no actual table of summed differences over displacements need be constructed. In practice, the algorithm has been shown to be effective both for synthetic and natural images using different types of features or measurements on the brightness data, including edges (both zero-crossings of the Laplacian of Gaussian and Canny's method), which generate sparse results along brightness edges, or brightness data directly, or the Laplacian of Gaussian, or its sign, which generate dense results. Because the optical flow is computed from quantities integrated over the individual patches, the results are robust against the effects of uncorrelated noise.

The comparison stage employs patchwise cross-correlation, which exploits local constancy of the optical flow (the velocity field is guaranteed to be constant for translations parallel to the image plane of a planar surface patch); it is a cubic polynomial for arbitrary motion of a planar surface (see Waxman, 1987; Little et al., 1987). Experimentally, we have used zero-crossings, the Laplacian of Gaussian filtered image, its sign, and the smoothed brightness values, with similar results. It is interesting that methods *superficially* so different (edge-based and intensity-based) give such similar results. As we mentioned earlier, this is not surprising. There are theoretical arguments that support, for instance, the equivalence of cross-correlating the sign bit of the Laplacian filtered

image and the Laplacian filtered image itself. The argument is based on the following theorem (see Little, Buelthoff, and Poggio, in preparation), which is a slight reformulation of a well-known theorem.

Theorem

If $f(x,y)$ and $g(x,y)$ are zero mean jointly normal processes, their cross-correlation is determined fully by the correlation of the sign of f and of the sign of g (and determines it). In particular

$$R_{\tilde{f},\tilde{g}} = \frac{2}{\pi} arcsin(R_{f,g})$$

where $\tilde{f} = sign\ f$ and $\tilde{g} = sign\ g$

Thus, cross-correlation of the sign bit is exactly equivalent to cross-correlation of the signal itself (for Gaussian processes). Notice that from the point of view of information, the sign bit of the signal is completely equivalent to the zero-crossing of the signal. Nishihara first used patchwise cross-correlation of the sign bit of DOG filtered images (Nishihara, 1984), and has implemented it more recently on real-time hardware (Nishihara and Crossley, 1988).

The existence of discontinuities can be detected in optical flow, as in stereo, both during computation and by processing the resulting flow field. The latter field is input to the MRF integration stage. During computation, discontinuities in optical flow arising from occlusions are indicated by low normalized scores for the chosen displacement.

2.4.4 Color

The color algorithm that we have implemented is a very preliminary version of a module that should find the boundaries in the surface spectral reflectance function, that is, discontinuities in the surface color. The algorithm relies on the idea of *effective illumination* and on the *single source* assumption, both introduced by Hurlbert and Poggio (see Poggio et al., 1985).

The single source assumption states that the illumination may be separated into two components, one dependent only on wavelength, and one dependent only on spatial coordinates; this generally holds for illumination from a single light source. It allows us to write the image irradiance equation for a Lambertian world as

$$I^{\nu} = k^{\nu} E(x,y) \rho^{\nu}(x,y)$$

where I^{ν} is the image irradiance in the νth spectral channel ($\nu = red, green, blue$), $\rho^{\nu}(x,y)$ is the surface spectral reflectance (or albedo), and the effective illumination $E(x,y)$ absorbs the spatial variations of the illumination and the shading due to the 3D shape of surfaces (k^{ν} is a constant

for each channel, and depends only on the luminant). A simple segmentation algorithm is then obtained by considering the equation

$$H(x,y) = \frac{I^r}{I^r + I^g} = \frac{k^r \rho^r}{k^r \rho^r + k^g \rho^g}$$

which changes only when ρ^r, or ρ^g, or both change. Thus H, which is piecewise constant, has discontinuities that mark changes in the surface albedo, independently of changes in the effective illumination.

The quantity $H(x,y)$ is defined almost everywhere, but is typically noisy. To counter the effect of noise, we exploit the prior information that H should be piecewise constant with discontinuities that are themselves continuous, non-intersecting lines. As we will discuss later, this restoration step is achieved by using a MRF model. This algorithm works only under the restrictive assumption that specular reflections can be neglected. Hurlbert (1989) discusses in more detail the scheme outlined here and how it can be extended to more general conditions.

2.4.5 Texture

The texture algorithm is a greatly simplified parallel version of the texture algorithm developed by Voorhees and Poggio (1987). Texture is a scalar measure computed by summation of texton densities over small regions surrounding every point. Discontinuities in this measure can correspond to occlusion boundaries, or orientation discontinuities, which cause foreshortening. Textons are computed in the image by simple approximation to the methods presented in Voorhees and Poggio (1987). For this example, the textons are restricted to blob-like regions, without regard to orientation selection.

To compute textons, the image is first filtered by a Laplacian of Gaussian filter at several different scales. The smallest scale selects the textural elements. The Laplacian of Gaussian image is then thresholded at a non-zero value to find the regions which comprise the blobs identified by the textons. The result is a binary image with non-zero values only in the areas of the blobs. A simple summation counts the density of blobs (the portion of the summation region covered by blobs) in a small area surrounding each point. This operation effectively measures the density of blobs at the small scale, while also counting the presence of blobs caused by large occlusion edges at the boundaries of textured regions. Contrast boundaries appear as blobs in the Laplacian of Gaussian image. To remove their effect, we use the Laplacian of Gaussian image at a slightly coarser scale. Blobs caused by the texture at the fine scale do not appear at this coarser scale, while the contrast boundaries, as well as all other blobs at coarser scales, remain. This coarse blob image filters the fine blobs; blobs at the coarser scale are removed from the fine scale image. Then, summation, whether with a simple scan operation, or Gaussian filtering, can determine the blob density at the fine scale

only. This is one example where multiple spatial scales are used in the present implementation of the Vision Machine.

2.4.6 The Integration Stage and MRF

Whereas it is reasonable that combining the evidence provided by multiple cues, for example, edge detection, stereo, and color, should provide a more reliable map of the surfaces than any single cue alone, it is not obvious how this integration can be accomplished. The various physical processes that contribute to image formation, *surface depth, surface orientation, albedo* (Lambertian and specular component), *illumination*, are coupled to the image data, and therefore to each other, through the imaging equation. The coupling is, however, difficult to exploit in a robust way, since it depends critically on the reflectance and imaging models. We argue that the coupling of the image data to the surface and illumination properties is of a more qualitative and robust sort at locations in which image brightness changes sharply and surface properties are discontinuous, in short, at edges. The intuitive reason for this is that at discontinuities, the coupling between different physical processes and the image data is robust and qualitative. For instance, a depth discontinuity usually originates a brightness edge in the image, and a motion boundary often corresponds to a depth discontinuity (and a brightness edge) in the image. This view suggests the following integration scheme for restoring the data provided by early modules. The results provided by stereo, motion, and other visual cues are typically noisy and sparse. We can improve them by exploiting the fact that they should be smooth, or even piecewise constant (as in the case of the albedo), between discontinuities. We can exploit *a priori* information about generic properties of the discontinuities themselves, for instance, that they usually are continuous and non-intersecting.

The idea is then to detect discontinuities in each cue, for instance depth, simultaneously with the approximation of the depth data. The detection of discontinuities is helped by information on the presence and type of discontinuities in the surfaces and surface properties (see Figure 1), which are coupled to the brightness edges in the image.

Notice that reliable detection of discontinuities is critical for a vision system, since discontinuities are often the most important locations in a scene; depth discontinuities, for example, normally correspond to the boundaries of an object or an object part. The idea is thus to couple different cues through their discontinuities and to use information from several cues simultaneously to help refine the initial estimation of discontinuities, which are typically noisy and sparse.

How can this be done? We have chosen to use the machinery of Markov Random Fields (MRFs), initially suggested for image processing by Geman and Geman (1984). In the following section, we will give a brief, informal outline of the technique and of our integration scheme. More detailed information about MRFs can be found in Geman and Geman (1984) and Marroquin et al.

(1987). Gamble and Poggio (1987) describe an earlier version of our integration scheme and its implementation as outlined in the next section.

MRF Models

Consider the prototypical problem of approximating a surface given sparse and noisy data (depth data) on a regular 2D lattice of sites. We first define the prior probability of the class of surfaces we are interested in. The probability of a certain depth at any given site in the lattice depends only upon neighboring sites (the Markov property). Because of the Clifford-Hammersley theorem, the prior probability is guaranteed to have the Gibbs form

$$P(f) = \frac{1}{Z} e^{-\frac{U(f)}{T}}$$

where Z is a normalization constant, T is called temperature, and $U(f) = \sum_C U_C(f)$ is an energy function that can be computed as the sum of local contributions from each neighborhood. The sum of the *potentials*, $U_C(X)$, is over the neighborhood's *cliques*. A clique is either a single lattice site or a set of lattice sites such that any two sites belonging to it are neighbors of one another. Thus $U(f)$ can be considered as the sum over the possible configurations of each neighborhood (see Marroquin et al.. 1987). As a simple example, when the surfaces are expected to be smooth, the prior probability can be given as sums of terms such as

$$U_c(f) = (f_i - f_j)^2$$

where i and j are neighboring sites (belonging to the same clique).

If a model of the observation process is available (i.e., a model of the noise), then one can write the conditional probability $P(g/f)$ of the sparse observation g for any given surface f. Bayes Theorem then allows one to write the posterior distribution

$$P(f/g) = \frac{1}{Z} e^{\frac{-U(f/g)}{T}}$$

In the simple earlier example, we have (for Gaussian noise)

$$U(f/g) = \sum_C \alpha \gamma_i (f_i - g_i)^2 + (f_i - f_j)^2$$

where $\gamma_i = 1$ only where data are available. More complicated cases can be handled in a similar manner.

The posterior distribution cannot be solved analytically, but sample distributions can be obtained using Monte Carlo techniques such as the Metropolis algorithm. These algorithms sample the space of possible surfaces according to the probability distribution $P(f/g)$ that is determined by

the prior knowledge of the allowed class of surfaces, the model of noise, and the observed data. In our implementation, a highly parallel computer generates a sequence of surfaces from which, for instance, the surface corresponding to the maximum of $P(f/g)$ can be found. This corresponds to finding the global minimum of $U(f/g)$ (simulated annealing is one of the possible techniques). Other criteria can be used: Marroquin (1985) has shown that the average surface f under the posterior distribution is often a better estimate, and one which can be obtained more efficiently by simply finding the average value of f at each lattice site.

One of the main attractions of MRFs is that the prior probability distribution can be made to embed more sophisticated assumptions about the world. Geman and Geman (1984) introduced the idea of another process, the line process, located on the dual lattice, and representing explicitly the presence or absence of discontinuities that break the smoothness assumption. The associated prior energy then becomes

$$U_C(f) = (f_i - f_j)^2(1 - l_i^j) + \beta V_C(l_i^j)$$

where l is a binary line element between site i,j. V_C is a term that reflects the fact that certain configurations of the line process are more likely than others to occur. In our world, depth discontinuities are usually themselves continuous, non-intersecting, and rarely isolated joints. These properties of physical discontinuities can be enforced locally by defining an appropriate set of energy values $V_C(l)$ for different configurations of the line process in the neighborhood of the site (notice that the assignment of zero energy values to the non-central cliques mentioned in Gamble and Poggio (1987) is wrong, as pointed out to us by Tal Symchony).

Organization of Integration

It is possible to extend the energy function to accommodate the interaction of more processes and their discontinuities. In particular, we have extended the energy function to couple several of the early vision modules (depth, motion, texture, and color) to brightness edges in the image. This is a central point in our integration scheme; brightness edges guide the computation of discontinuities in the physical properties of the surface, thereby coupling surface depth, surface orientation, motion, texture, and color, each to the image brightness data and to each other. The reason for the role of brightness edges is that changes in surface properties usually produce large brightness gradients in the image. It is exactly for this reason that edge detection is so important in both artificial and biological vision.

The coupling to brightness edges may be done by replacing the term $V_C(l_i^j)$ in the last equation with the term

$$V(l,e) = g(e_i^j, V_C(l_i^j))$$

with e_i^j representing a measure of the presence of an brightness edge between site i,j. The term

g has the effect of modifying the probability of the line process configuration depending on the brightness edge data ($V(l,e) = -log\ p(l/e)$). This term facilitates formation of discontinuities (that is, l_i^j) at the locations of brightness edges. Ideally, the brightness edges (and the neighboring image properties) activate, with different probabilities, the different surface discontinuities (see Figure 1), which in turn are coupled to the output of stereo, motion, color, texture, and possibly other early algorithms.

We have been using the MRF machinery with prior energies like that shown above (see also Figure 1) to integrate edge brightness data with stereo, motion, and texture information on the MIT Vision Machine System.

We should emphasize that our present implementation represents a subset of the possible interactions shown in Figure 1, itself only a simplified version of the organization of the likely integration process. The system will be improved in an incremental fashion, including pathways not shown in Figure 1, such as feedback from the results of integration into the matching stage of the stereo and motion algorithms.

Algorithms: Deterministic and Stochastic

We have chosen to use MRF models because of their generality and theoretical attractiveness. This does not imply that stochastic algorithms must be used. For instance, in the cases in which the MRF model reduces to standard regularization (Marroquin et al., 1987) and the data are given on a regular grid, the MRF formulation leads not only to a purely deterministic algorithm, but also to a convolution filter. Recent work in color (Hurlbert and Poggio, 1989) shows that one can perform integration similar to the MRF-based scheme using a deterministic update. Geiger and Girosi (1989) have shown that there is a class of deterministic schemes that are the mean-field approximations of the MRF models. These schemes have a much higher speed than the Montecarlo schemes we used so far, while promising similar performance.

2.5 Illustrative Results

Figures 2 and 3 show the results of the Vision Machine applied to the scene in Figure 2 and some of the intermediate steps. Figure 3 shows the brightness edges computed by the Canny algorithm at two different spatial scales ($\sigma = 2.5$ and $\sigma = 4$). We show neither the stereo pair nor the motion sequence in which the teddy bear was rolling slightly on his back from one frame to the next. The results given by the stereo, motion, texture and color algorithms, after an initial smoothing to make them dense (see Gamble and Poggio, 1987), are shown in the first column on the left of Figure 4 (from top to bottom). They represent the input to the MRF machinery that integrates each of those data sets with the brightness edges. The color algorithm uses the edges at the coarser resolution, since we want to avoid detecting texture marks on the surface; the other cues

Figure 2: Grey-level image of a natural scene processed by the Vision Machine.

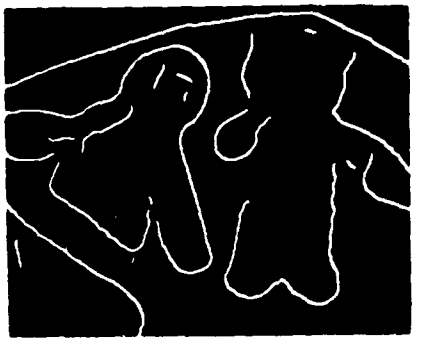

Figure 3: Canny edges of the images in Figure 2.

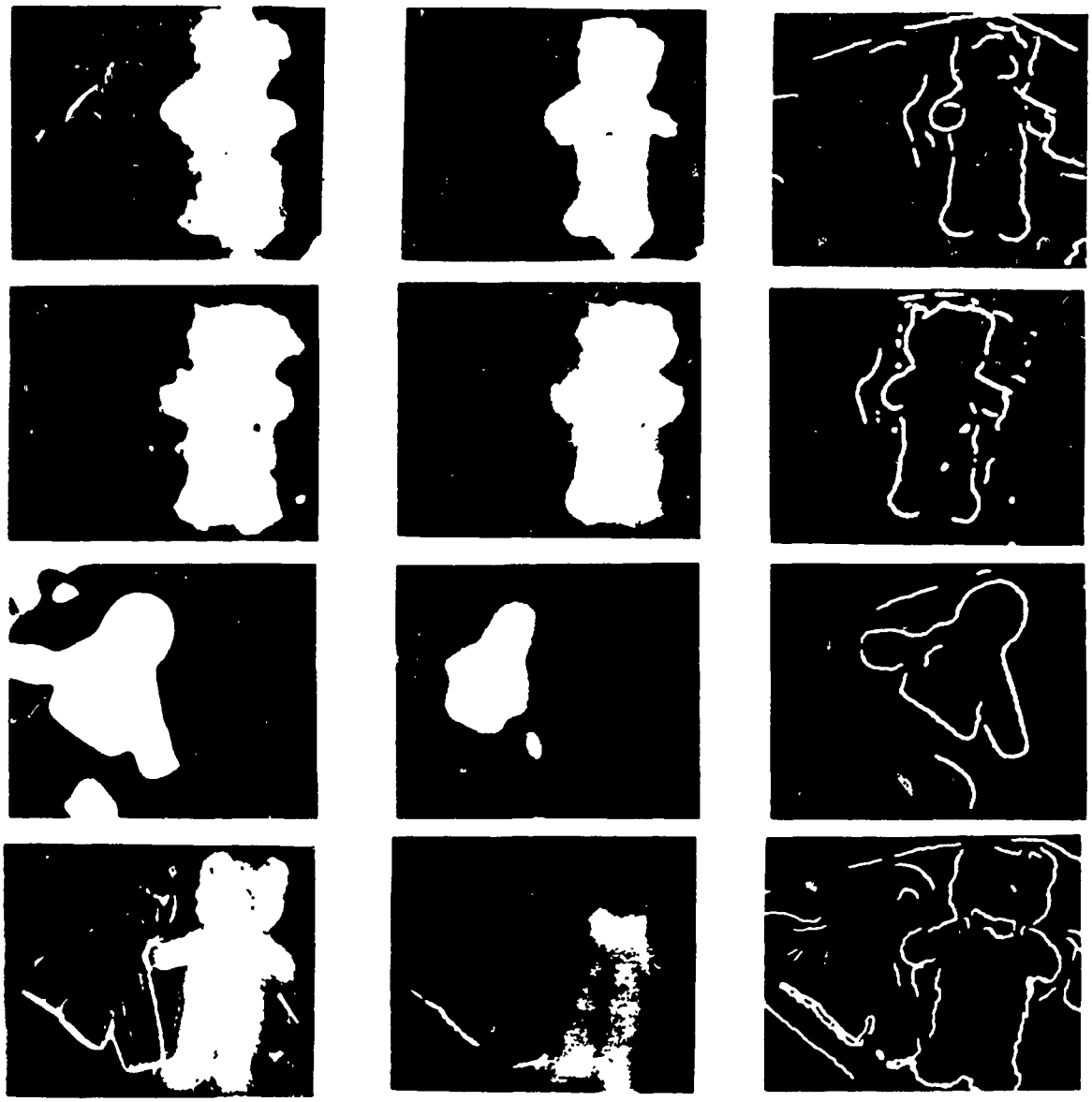

Figure 4: MRF results for stereo, motion, texture, and color.

Figure 5: Union of depth and motion discontinuities.

are integrated with the Canny edges at a smaller scale ($\sigma = 2.5$). The central column of Figure 4 shows the reconstructed depth, color (the quantity H defined earlier), texture and motion flow; the left column show the discontinuities found by the MRF machinery in each of the cues. Processing of the stereo output finds depth discontinuities in the scene (mainly the outlines of the teddy, plus a fold of a wet suit protruding outward). Motion discontinuities are found by the MRF machinery with help from brightness edges. The color boundaries show regions of constant surface color, independently of its shading: notice, for instance, that brightness edges inside the teddy bear, due to shading, do not appear as color edges (the color images were taken from a different camera). The texture boundaries correspond quite well to different textured surfaces.

Figure 5 shows that the union of the discontinuities in depth and motion for the scene of Figure 2 gives a rather good "cartoon" of the original scene. At the same time, our integration algorithm achieves a preliminary classification of the brightness edges in the image, in terms of their physical origin. A more complete classification will be achieved by the full scheme in : the lattices at the top classify the different types of discontinuities in the scene. The set of such discontinuities in the various physical processes should represent a good set of data for later recognition stages.

2.6 Recognition

The output of the integration stage provides a set of edges labeled in terms of physical discontinuities of the surface properties. They represent a good input to a model-based recognition algorithm like the ones described by Dan Huttenlocher and Todd Cass in the 1988 Proceedings of the Image Understanding Workshop. In particular, we have interfaced the Vision Machine as implemented so far with the Cass algorithm. We have used only discontinuities for recognition; later we will also use the information provided by the MRFs on the surface properties *between* discontinuities.

We have more ambitious goals for the recognition stage of the Vision Machine. In an unconstrained environment the library of models that a system with human-level performance requires is in the order of many thousands. Thus, the ability to learn from examples appears to be essential for the achievement of high performance in real-world recognition tasks. Learning the models becomes then a primary concern in developing a recognition system for the Vision Machine. This has not been the case in other approaches of the last few years, mainly motivated by a robotic framework.

2.6.1 Learning in a three-stage recognition scheme

Although some of the existing recognition systems incorporate a module for learning object models from examples (e.g., Tucker's 2D system [67]) no such capability exists yet for the more difficult problems of recognizing 3D objects [37] or handwriting [16]. We believe that incorporating learning into a general-purpose recognition system may be facilitated by breaking down the task of

recognition into three distinct but interacting stages: *selection*, *indexing* and *verification*.

Selection

Selection or segmentation breaks down the image into regions that are likely to correspond to single objects. The utility of an early segmentation of a scene into meaningful entities lies in the great reduction of complexity of scene interpretation. Each of the detected objects can in turn be subjected to separate recognition, by comparing it with object models stored in memory. Without prior segmentation, every possible combination of image primitives such as lines and blobs can in principle constitute an object and must be checked out. The power of early segmentation may be enhanced by integrating all available visual cues, especially if the integration parameters are automatically adjusted to suit the particular scene in question.

Indexing

By indexing we mean defining a small set of candidate objects that are likely to be present in the image. Although one cannot hope to achieve an ideal segmentation in real-world situations, partial success is sufficient if the indexing process is robust. Assuming that most objects in the real world are redundantly specified by their local features, a good indexing mechanism would use such features to overcome changes in viewpoint and illumination, occlusion and noise.

What kind of feature is good for indexing? Reliably detected lines provided by the integration of several low-level cues in the process of segmentation may suffice in many cases. We conjecture that *simple* viewpoint-invariant combinations of primitive elements, such as two lines forming a corner, parallel lines and symmetry are also likely to be useful. Ideally, only 2D information should be used for indexing, although it may be augmented sometimes by qualitative 3D cues such as relative depth.

Verification

In the verification stage each of the candidates screened by the indexing process is tested to find the best match to the image. At this stage, the system can afford to perform complicated tests, since the number of candidate objects is small. We conjecture that hierarchical indexing by a small number (two or three) features that are spatially localized in 2D suffices to achieve useful interpretations of most everyday scenes. In general, however, further verification by task-dependent routines [68] or precise shape matching, possibly involving 3D information, is required [69] [47] [37][67] [7] [1].

2.7 Future Developments

The Vision Machine should evolve in several parallel directions:

- improvement and extensions of its early modules
- improvement of the integration and recognition stages (recognition is discussed later)
- use of the eye-head system in an active mode during recognition task by developing appropriate gaze strategies
- use of the results of the integration stage in order to improve the operation of early modules such as stereo and motion by feeding back the preliminary computation of the discontinuities

Two goals will occupy most of our attention, if we will be able to continue to work on the project. The first one is the development of the overall organization of the Vision Machine. The system can be seen as an implementation of the *inverse optics* paradigm: it attempts to extract surface properties from the integration of image cues. It must be stressed that we never intended this framework to imply that precise surface properties such as dense, high resolution depth maps, must be delivered by the system. This extreme interpretation of inverse optics seems to be common, but was not the motivation of our project, which originally started with the name *Coarse Vision Machine* to emphasize the importance of computing qualitative, as opposed to very precise, properties of the environment.

Our second main goal in the Vision machine project will be Machine Learning, that we will discuss in the next chapter. In particular, we have begun to explore simple learning and estimation techniques for vision tasks. We have succeded in synthetizing a color algorithm from examples [36] and in developing a technique to perform unsupervised learning [63] of other simple vision algorithms such as simple versions of the computation of texture and stereo. In addition, we have used learning techniques to perform integration tasks, such as labeling the type of discontinuities in a scene. We have also begun to explore the connections between recent approaches to learning, such as neural networks, genetic algorithms, and classical methods in approximation theory such as splines, Bayesian techniques and Markov Random Field models, as discussed in one of the next chapters. We have identified some common properties of all these approaches and some of the common limitations, such as sample complexity. As a consequence, we now believe that we can leverage our expertise in approximation techniques for the problem of learning in machine vision.

For further details and background information on this work, see the following refernces: [75, 60, 3, 46, 39, 32, 74, 43, 71, 8, 49, 10, 41, 64, 65, 40, 59, 58, 70].

3 VLSI

3.0.1 A VLSI Vision Machine?

Our Vision Machine consists mostly of specialized software running on a general purpose computer. the Connection Machine. This is a good system for the present stage of experimentation and development. Later, once we have perfected and tested the algorithms and the overall system, it will make sense to compile the software in silicon in order to produce a faster, cheaper, and smaller Vision Machine. We are presently planning to use VLSI technologies to develop some initial chips as a first step toward this goal. In this section, we will outline some thoughts about VLSI implementation of the Vision Machine.

Algorithms and Hardware

We realize that our specialized software vision algorithms are not, in general, optimized for hardware implementation. So, rather than directly "hardwiring algorithms" into standard computing circuitry, we will be investigating "algorithmic hardware" designs that utilize the local, symmetric nature of early vision problems. This will be an iterative process, as the algorithm influences the hardware design and as hardware constraints modify the algorithm.

Degree of Parallelism

Typical vision tasks require tremendous amounts of computing power, and are usually parallel in nature. As an example, biological vision uses highly parallel networks of relatively slow components to achieve sophisticated systems. However, when implementing our algorithms in silicon integrated circuits, it is not clear what level of parallelism is necessary. While biology is able to use three dimensions to construct highly interconnected parallel networks, VLSI is limited to $2\frac{1}{2}$ dimensions, making highly parallel networks much more difficult and costly to implement. However, the electrical components of silicon integrated circuits are approximately four orders of magnitude faster than the electrochemical components of biology. This suggests that pipelined processing or other methods of time-sharing computing power may be able to compensate for the lower degree of connectivity of silicon VLSI. Clearly, the architecture of a VLSI vision system may not resemble any biological vision systems.

Signal Representation

Within the integrated circuit, the image data may be represented as a digital word or an analog value. While the advantages of digital computation are its accuracy and speed, digital circuits do not have as high a degree of functionality per device as analog circuits. Therefore, analog circuits should allow much denser computing networks. This is particularly important for the integration of computational circuitry and photosensors, which will help to alleviate the I/O bottleneck typically experienced whenever image data are serially transferred between Vision Machine components.

However, analog circuits are limited in accuracy, and are difficult to characterize and design.

The primary motivation for a VLSI implementation of our Vision Machine is to increase the computational speed and reduce the physical size of the components, with the eventual goal of real-time, mobile vision systems. While the main computational engine of our Vision Machine is the Connection Machine, which is a very powerful and flexible SIMD computer, specific VLSI implementations will attempt to tradeoff computational flexibility for faster performance and higher degree of integration. A VLSI implementation of our Vision Machine can offer significant improvements in performance that would be difficult or impossible to attain by other methods. Presently, we are specifically investigating the integration of charge coupled devices for photosensing and simple parallel computations, such as binomial convolution and patchwise correlation. In particular, Woody Yang has developed and fabricated CCDs circuits for signal processing and imaging, described some basic operations and how those operations can be combined into a CCD processor architecture for vision. A circuit for performing Laplacian-of-Gaussian filtering of the image has been sent to fabrication. The paper discusses other CCD circuits for the integration-reconstruction stage of the Vision Machine and for stereo.

4 Learning

Poggio and Girosi have recently obtained what we believe is a satisfactory understanding of the learning obtained by "neural" networks such as backpropagation. In the last Proceedings we had drawn a formal analogy between simple forms of learning and hypersurface reconstruction. As a consequence, learning can be achieved by techniques such as regularization and therefore generalized splines. The connection, however, between these classical methods and feedforward networks of the backpropagation type remained unclear. Poggio and Girosi have now found that the missing link is provided by the approximation method of Radial Basis Functions. The Radial Basis Function approximation method has a sound theoretical basis and a direct interpretation in term of a feedforward network with one "hidden" layer. Poggio and Girosi have been able to prove its connections to generalized splines, to regularization techniques and to Bayes' approaches. They have developed several new extensions of the method and indicated how to address a few general issues in networks and learning within its formal framework (Girosi and Poggio, 1989, 1990) .

We describe briefly the interpolation and approximation technique called Radial Basis Functions, which has been used in the past for surface interpolation with very promising results; clearly surface reconstruction is another application of this technique of interest to vision research.

4.1 Radial Basis Functions

Given a set $D = \{(\vec{x}_i, \vec{y}_i) \in R^n \times R | i = 1....N\}$ of data to interpolate, the Radial Basis Function method corresponds to choosing the form of the interpolating function as

$$F(\vec{x}) = \sum_{i=1}^{N} c_i h(\|\vec{x} - \vec{x}_i\|^2)$$

where h is a smooth univariate function defined on $[0, \infty)$ and $\|\cdot\|$ is a norm on R^n. This formula means that the interpolating function is expanded on a finite N-elements basis that is given from the set of functions h translated and centered at data points. The N unknown coefficients of the expansion can be recovered imposing the interpolating conditions $F(\vec{x}_i) = Y_i$. This gives the linear system

$$Y_j = \sum_{i=1}^{N} c_i h(\|\vec{x}_j - \vec{x}_i\|^2) \quad j = 1, ..., N.$$

Defining the vectors $\vec{Y}$, $\vec{c}$ and the symmetric matrix H as follows

$$(\vec{Y})_i = Y_i, \qquad (\vec{c})_i = c_i, \qquad (H)_{ij} = h(\|\vec{x}_j - \vec{x}_i\|^2)$$

we obtain

$$\vec{c} = H^{-1}\vec{Y}$$

provided H is invertible. The invertibility of H depends on the choice of the function h. In fact Micchelli proved the following theorem, that defines a class of functions that we can choose to form the basis:

Theorem 4.1.1 *Let G be a continuous function on $[0, \infty)$ and positive on $(0, \infty)$. Suppose its first derivative is completely monotonic but not constant on $(0, \infty)$. Then for any distinct vectors $\vec{x}_1, ..., \vec{x}_N \in R^n$*

$$(-1)^{n-1} det G(\|\vec{x}_i - \vec{x}_j\|^2) > 0$$

The interpolation conditions can be weakened if the number of knots is made lower than the number of data and their coordinates are allowed to be chosen arbitrarily. In this case, denoting with $\vec{t}_1, ..., \vec{t}_K$ the coordinates of the K knots ($K < N$) the interpolation conditions give the linear system $\vec{Y} = H\vec{c}$ where $(H)_{i\alpha} = h(\|\vec{x}_i - \vec{t}_\alpha\|^2)$ ($i = 1, ..., N$ and $\alpha = 1, ..., K$). The matrix H being rectangular ($N \times K$), this system is overconstrained and the problem must be then regularized

to obtain a reasonable set of coefficients for the expansion. A least-squares approach can then be adopted and the optimal solution can be written as

$$\vec{c} = H^{+}\vec{Y}$$

where H^{+} is the Moore-Penrose pseudo-inverse. In the overdetermined case, one has

$$H^{+} = (H^{T}H)^{-1}H^{T}.$$

As in the previous case this formulation makes sense if the matrix $H^{T}H$ is non singular. Micchelli's theorem is still relevant to this problem, since Poggio and Girosi proved the following corollary:

Theorem 4.1.2 *Let G be a function satisfying the conditions of Micchelli's theorem and $\vec{x}_1, ..., \vec{x}_N$ a N-tupla of vectors in R^n. If H is the $(N-s) \times N$ matrix H obtained from the matrix $G_{i,j} = G(\|\vec{x}_i - \vec{x}_j\|^2)$ deleting s arbitrary rows, then the $(N-s) \times (N-s)$ matrix $H^{T}H$ is not singular.*

The first layer consists of "input" units whose number is equivalent to the number of independent variables of the problem. The second layer implements the set of radial basis function and its number of units is equal to the number of knots. The units of the second layer are in general fully connected to the units of the first one. The third layer consists of one unit (for a scalar function) connected to all the units of the second layer and computing a weighted sum of their outputs. The weights are the coefficients of the radial basis expansion and are the only unknown of the problem. Since spline interpolation can be implemented by such a network, and spline are known to have a large power of approximation we have then shown that a high degree of approximation can be obtained by just one hidden layer network.

4.2 An extension: Generalized Radial Basis Functions

Poggio and Girosi noticed that the knots of the radial basis expansion have been kept fixed, the weights being the only unknowns. To make the method more flexible they propose to consider even the knots as unknowns and to look for the configuration of weights *and* knots that minimizes the least square error on the data. The problem consists then in finding the values of the coefficients c_i and knots $\vec{t}_\alpha$ that minimizes the function

$$E = \sum_{i=1}^{N}(Y_i - \sum_{\alpha=1}^{K} c_\alpha h(\|\vec{x}_i - \vec{t}_\alpha\|^2))^2.$$

A gradient-descent approach can be adopted to find the solution to this problem. The values of c_α and $\vec{t}_\alpha$ are then regarded as the coordinates of the stable fixed point of the following dynamical system:

$$\dot{c_\alpha} = -\omega \frac{\partial E}{\partial c_\alpha}, \quad \alpha = 1, ...K$$

$$\dot{\vec{t}_\alpha} = -\omega \frac{\partial E}{\partial \vec{t}_\alpha}, \quad \alpha = 1, ...K$$

where ω is a parameter determining the microscopic timescale of the problem and is related to the rate of convergence to the fixed point. Defining the interpolation error as

$$\Delta_i = Y_i - \sum_{\alpha=1}^{K} c_\alpha h(\|\vec{x}_i - \vec{t}_\alpha\|^2)$$

we can write the gradient terms as

$$\frac{\partial E}{\partial c_\alpha} = -2 \sum_{i=1}^{N} \Delta_i h(\|\vec{x}_i - \vec{t}_\alpha\|^2) \ ,$$

$$\frac{\partial E}{\partial \vec{t}_\alpha} = 4 c_\alpha \sum_{i=1}^{N} \Delta_i h'(\|\vec{x}_i - \vec{t}_\alpha\|^2)(\vec{x}_i - \vec{t}_\alpha)$$

where h' is the first derivatives of h. Equating $\frac{\partial E}{\partial \vec{t}_\alpha}$ to zero we notice that at the fixed point the knot vectors $\vec{t}_\alpha$ satisfy the following equation:

$$\vec{t}_\alpha = \frac{\sum_i P_i^\alpha \vec{x}_i}{\sum_i P_i^\alpha}$$

where $P_i^\alpha = \Delta_i h'(\|\vec{x}_i - \vec{t}_\alpha\|^2)$. The optimal knots are then a weighted sum of the data points. The weight P_i^α of the data point i for a given knot α is high if the interpolation error Δ_i is high there *and* the radial basis function centered on that knot changes quickly in a neighbor of the data point.

4.3 RBF are equivalent to regularization

Interesting connections between RBF and regularization techniques arise when the basis function are chosen to be Gaussian. Let us consider the RBF method in its original formulation, having chosen the basis function to be a Gaussian G. The coefficients of the expansion are the solution of the linear system $\vec{Y} = G\vec{c}$ where $(G)_{ij} = G(\|\vec{x}_i - \vec{x}_j\|^2)$. If data are noisy a well known technique [65] to regularize the solution is to substitute the previous linear system with the following

$$\vec{Y} = (G + \lambda I)\vec{c}$$

where λ is a small parameter and I is the identity matrix. We now show that the same approximating function can be obtained from a pure regularization approach. Let us consider the following functional

$$E_1[F] = \sum_i (Y_i - F(\vec{x}_i))^2 + \lambda \int d\vec{x} \sum_{m=0}^{\infty} a_m (D^m F(\vec{x}_i))^2$$

where λ is a parameter, $D^{2m} = \nabla^{2m}$, $D^{2m+1} = \vec{\nabla}\nabla^{2m}$, ∇^2 is the Laplacian operator and the coefficients a_m are to be chosen. It can be easily proved that by posing $a_m = \frac{\sigma^{2m}}{m!2^m}$ the function that minimizes this functional can be written as

$$F(\vec{x}) = \sum_{i=1}^{N} c_i G(\|\vec{x} - \vec{x}_i\|^2) \tag{1}$$

where G is a Gaussian of variance σ and the coefficients satisfy the linear system $\vec{Y} = (G + \lambda I)\vec{c}$, that is the same as before. So in this case RBF and regularization are equivalent. Notice that changing the coefficients a_m is equivalent to selecting another basis function h instead of G. In fact it can be shown that the set a_m and h are related by the following distributional partial differential equation:

$$\sum_{m=0}^{\infty} (-1)^m a_m \nabla^{2m} h(\vec{x}) = \delta(\vec{x}) \ .$$

The stabilizer described above is not the most general one. Other types could have been chosen, depending on the a priori information about the surface to be reconstructed. The previous one is suitable if we want to keep local the interaction between a data point and its neighbors, since the Gaussian falls off very quikly, that is the "interaction" is short range. It can be shown that this is related to the presence of a term of degree zero in the stabilizer. For example, in two dimensions, if we chose a stabilizer like

$$\int dxdy \left[\left(\frac{\partial^2 F}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 F}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 F}{\partial y^2} \right)^2 \right]$$

this leads to a Radial Basis Function of the type $h(\|\vec{x}\|^2) = \|\vec{x}\|^2 log\|\vec{x}\|$. This kind of interaction is clearly long-range, as it should be, since the corresponding functional is the bending energy of a thin plate of infinite extent (Duchon and Meinguet gave the name *thin plate splines* to the solution of the interpolation problem obtained minimizing this functional).

The same kind of results can be obtained in a third way, in the networks framework. Let us consider the network and the problem of finding the "synaptic" weights. I we adopt a least square criterion we recover the usual linear system $\vec{Y} = G\vec{c}$, but often it is considered an advantage to keep the connections from growing to infinity, and so the following functional is minimized:

$$E_2[F] = \sum_i (Y_i - \sum_{i=1}^{N} c_i G(\|\vec{x} - \vec{x}_i\|^2))^2 + \lambda \sum_i c_i^2$$

where the last term gives an high price to the configurations in which some coefficient c_i is very high. It is immediate to see that the minimization of this functional leads to the solution of the linear system $\vec{Y} = (G + \lambda I)\vec{c}$. This shows the equivalence between some of the "new" neural networks techniques and classical regularization.

5 Other Work

5.1 Labeling the physical origin of edges: computing qualitative surface attributes

Physical Discontinuities

We classify edges according to the following physical events: discontinuities in surface properties, called *mark* or *albedo* edges (e.g., changes in the color of the surface); discontinuities in the orientation of the surface patch, called *orientation* edges (e.g., an edge in a polyhedron); discontinuities in the illumination, called *shadow* edges; *occluding boundaries*, which are discontinuities in the object space (a different object); and *specular* discontinuities, which exist for non-Lambertian objects.

Gamble, Geiger, Poggio, and Weinshall have implemented a part of the general scheme [18]. More specifically, they have used a simple linear classifier to label edges at pixels where there exists an intensity discontinuity, using the output of the line process associated with each low-level vision module. They use the fact that the modules' discontinuities are aligned, having being integrated with the intensity edges before, so that the nonexistence of a module discontinuity at a pixel is meaningful. The linear classifier corresponds to a linear network where each output unit is a weighted linear combination of its inputs (for a similar application to a problem of color vision, see [36]). The input to the network is a pixel where there exists an intensity edge and that feeds a set of qualitatively different input units. The output is a real value vector of labels' support.

In the system we have so far implemented, we achieve a rather restricted integration, since each module is integrated only with the intensity module, and labeling is done via a simple linear classifier only. It is still unclear how successfull labeling can be, using only local information.

5.2 Saliency, grouping and segmentation

A grouping and segmentation module working on the output of the edge detection module is an important part of a vision system: humans can deal with monocular, still, black and white pictures devoid of stereo, motion and color. We are now developing techniques to find salient edges, to group them and thereby segment the image. These algorithms have not been integrated yet in the Vision Machine system.

5.2.1 Saliency Measure

Edge maps produced by most current edge detectors are cluttered with edge responses and may have edges caused by noise. This creates difficulties for higher level processing, since the combinatorics of these algorithms often depends on the number of edge primitives being examined. What is needed is a technique to focus attention on the "important" edges in a scene. We call such attention focusing techniques that measure the "importance" of an edge saliency measures. Shimon Ullman has proposed two different kinds of saliency measures: local saliency and structural saliency. An edge's local saliency is entirely determined by features of that edge alone. For example, an edge's length, its average gradient magnitude, or the color of a bounding region serve as local saliency measures. Structural saliency refers to more global properties of an edge – its relationships with other edges. Although two edges may not be locally salient, if there is a "nonaccidental" relationship between them, then the structure becomes salient. Examples of "nonaccidental" relationships, as pointed out by David Lowe, include collinearity, parallelism, and symmetry, among other things.

We have investigated local saliency measures applied to the output of the Canny edge detector (Beymer, in preparation). The edge features we have considered include curvature, edge length, and gradient magnitude. The measure favors those edges that have low average curvature, long length, and a high gradient magnitude. The saliency measure eliminates many of the edges due to noise and many of the unimportant edges. The edges that remain are often the long, smooth boundaries of objects and significant intensity changes inside the objects. We expect that the salient edges will help higher level processes such as grouping (structural saliency) and model based recognition by allowing them to focus attention on regions of an image bounded by salient edges.

5.2.2 T Junctions: Their Detection and Use in Grouping

In cluttered imagery, imagery containing many objects occluding one another, it is important to group together pieces of the image that come from the same object. In particular, given an edge map produced by the Canny edge detector, we would like to select and group together the edges from a particular object before running high level recognition algorithms on the edge data. This

grouping stage helps reduce the combinatorics of the higher level stages, as they are not forced to consider false edge groupings as objects. Considering how occlusion cues can be used in grouping, we have investigated the detection of T junctions and grouping rules arising from the pairing of T junctions. When one object partially occludes another in a cluttered scene, a T junction is formed between the two objects. David Beymer has developed algorithms for detecting T junctions as a postprocessing step to the Canny edge detector. The Canny edge detector, while very good at detecting edges, is particularly bad at detecting junctions. Indeed, it was designed to detect one dimensional events. This one dimensional characterization of the image breaks down at junctions since locally there are three or more surfaces in the image. We have investigated how one could use edge curvature and region properties of the image to reconstruct these "broken" junctions. Often the way Canny will fail at junctions is that one of the three curves belonging to the junction will be broken off from the other two. We have modified an existing algorithm asnd achieved promisisng results in restoring broken T junctions. Once located in the image, T junctions are represented by three edges, the left part of the top horizontal edge of the T, the right part, and the stem. The top horizontal edges are the occluding edges and the vertical stem is the occluded edge. Given the junctions, we can start pairing T junctions and grouping edge fragments. If we assume that all objects in the scene fit entirely within the image boundaries, all T junctions must be matched up with a "brother" T junction along the occluded edge joining them. This constraint helps to classify T junctions, making their detection more robust. Once a T junction is matched with its brother, we know exactly which edge is the occluded edge (it is the edge that is traced to reach the brother), so we can group the two occluding edges together. The occluded edge will be extended, starting a search process to bridge the occluding object. Here we are looking for an opposing T junction on the other side of the occluding object. If such a pair of opposing Ts is found, we can group together the occluded edges of the respective T junctions. The application of these grouping rules for occluding and occluded edges often product closed contours when the Canny edges are fairly good. For each closed contour, we can form a closed region corresponding to an object or object part in the image. Finally, the T junctions are used to calculate relative depth information among the regions. In the end, the system can divide the image into regions corresponding to objects and give their relative depths. The algorithm is presently working on "toy" images made from construction paper cutouts and has not been integrated in the Vision Machine system.

5.3 Fast Vision: The Role of Time Smoothness

The present version of the Vision Machine processes only isolated frames. Even our motion algorithm takes as input simply a sequence of two images. The reason for this is, of course, limitations in raw speed. We cannot perform all of the processing we do at video rate (say, 30 frames per second), though this goal is certainly within present technological capabilities. If we could process

frames at video rate, we could exploit constraints in the time dimension similar to the ones we are already exploiting in the space domain. Surfaces, and even the brightness array itself, do not usually change too much from frame to frame. This is a constraint of smoothness in time, which is valid almost everywhere, but not *across* discontinuities in time. Thus one may use the same MRF technique, applied to the output of stereo, motion, color, and texture, and enforce continuity in time (if there are no discontinuities), that is, exploit the redundancy in the sequence of frames.

We believe that the surface reconstructed from a stereo pair usually does not need to be recomputed completely when the next stereo pair is taken a fraction of a second later. Of course, the role of the MRFs may be accomplished in this case by some more specific and more efficient deterministic method such as, for example, a form of Kalman filtering. Notice that space-time MRFs applied to the brightness arrays would yield spatiotemporal interpolation and approximation of a kind already considered (Fahle and Poggio, 1980; Poggio, Nielsen, and Nishihara, 1982; Bliss, 1985).

5.4 Parameter Estimation in the MRF integration stage

Using the MRF model involves an energy function which has several free parameters, in addition to the many possible neighborhood systems. The values of these parameters determine a distribution over the configuration-space to which the system converges, and the speed of convergence. Thus rigorous methods for estimating these parameters are essential for the practical success of the method and for meaningful results. In some cases, parameters can be learned from the data: e.g., texture parameters (Geman and Graffigne, 1987), or neighborhood parameters (for which a cellular automaton model may be the most convenient for the purpose of learning). There are general statistical methods which can be used for parameter estimation:

- A maximum likelihood estimate – one can use the indirect iterative EM algorithm (Dempster et.al., 1977), which is most useful for maximum likelihood estimation from incomplete data (see Marroquin, 1987 for a special case). This algorithm involves the iterative maximization (over the parameter space) of the expected value of the likelihood function given that the parameters take the values of their estimation in the previous iteration. Alternatively, a search constrained by some statistics for a minimum of an appropriate merit function may be employed (see Marroquin, 1987).

- A smoothing (regularization) parameter can be estimated using the methods of cross-validation or unbiased risk, to minimize the mean square error. In cross-validation, an estimate is obtained omitting one data point. The goal is to minimize the distance between the predicted data point (from the estimate above with the point omitted) and the actual value, for all points.

In the case of Markov Random Fields, some more specific approaches are appropriate for parameter estimation:

1) Besag (1972) suggested conditional maximum likelihood estimation using coding methods, maximum likelihood estimation with unilateral approximations on the rectangular lattice, or "maximum pseudolikelihood" – a method to estimate parameters for homogeneous random fields (see Geman and Graffigne, 1987).

2) For the MPM estimator, where a fixed temperature is yet another parameter to be estimated, one can try to use the physics behind the model to find a temperature with as little disorder as possible and still reasonable time of convergence to equilibrium (e.g., away from "phase-transition").

An alternative asymptotic approach can be used with smoothing (regularization) terms: instead of estimating the smoothing parameter, let it tend to 0 as the temperature tends to 0, to reduce the smoothing close to the final configuration (see Geman and Geman, 1987).

In summary, we plan to explore three distinct stages for parameter estimation in the integration stage of the Vision Machine:

- *Modeling* (from the physics of surfaces, of the imaging process and of the class of scenes to be analyzed and the tasks to be performed) and the form of the prior and of some conditional probabilities involved (e.g., the type of physical edges from properties of the measurements, such as characteristics of the brightness data). Range of allowed parameter values may also be established at this stage (e.g., minimum and maximum brightness value in a scene, depth differences, positivity of certain measurements, distribution of expected velocities, reflectance properties, characteristics of the illuminant, etc.).
- *Estimating* of parameter values from set of examples in which data and desired solution are given. This is a *learning stage.* We may have to use days of CM time and, at least initially, synthetic images to do this.
- *Tuning* of some of the parameters directly from the data (by using EM algorithm, cross-validation, Besag's work, or various types of heuristics).

The dream is that at some point in the future the Vision Machine will run all the time, day and night, looking about and learning on its own to see better and better.

5.5 Object Recognition

In earlier reports, we have described a series of approaches to the problem of model-based object recognition, based on matching object shape. Our work has proceeded along a number of fronts.

5.5.1 Recognition from Matched Dimensionalities

Earlier reports described the work of Grimson and Lozano-Pérez on the recognition of occluded objects from noisy sensory data under the condition of matched dimensionality [29]. Specifically, if the objects to be recognized and localized are laminar and lie on a flat surface, or if the objects are volumetric but lie in stable configurations on a flat surface, then the sensory data need only be two-dimensional (e.g. a single image); if the objects to be recognized and localized are volumetric and lie in arbitrary positions, then the sensory data must be three dimensional (e.g. stereo or motion data, laser range data). The original technique (called RAF) was designed to recognize polyhedral objects from simple measurements of the position and surface orientation of small patches of surface. The technique searches for consistent matchings between the faces of the object models and the sensory measurements, using constraints on the relative shape of pairs of model faces and pairs of measurements to reduce the search.

Our empirical work on RAF has advanced along a number of dimensions. First, we have shown that the RAF framework can successfully recognize and locate objects based on a variety of geometric features: edges, vertices, curved arcs, planar surface patches, and axes of cylinders and cones. Second, we have also shown that such features can be extracted from a range of sensory information, including grey level images, stereo data, motion data, sonar returns, laser striping data and tactile data. Third, we have shown that the RAF framework can be extended to deal with some classes of parameterized objects. These include the recognition of objects that can scale in size, the recognition of objects that are composed of rigid subparts connected through rotational degrees of freedom (e.g. a pair of scissors) and the recognition of objects that can undergo a stretching deformation along one axis.

Our empirical experience with RAF suggested that the method was remarkably efficient when dealing with data from a single object, but was inefficient when spurious data was included. To overcome this, we have incorporated a Hough transform to preselect portions of the search space on which to focus attention, and we have used thresholds on the goodness of an interpretation to terminate search. The combination of these two techniques resulted in dramatic improvement in the efficiency of the search method. Based on these observations, we have been developing a formal basis for explaining these results. In particular, we have shown the following formal results:

- If all of the data is known to have come from a single object, the expected amount of search is quadratic in the number of data and model features.
- If spurious data is included, the expected amount of search is a combination of polynomial in the number of data and model features, but exponential in the size of the actual correct interpretation.
- Using a Hough transform to preselect subspaces of the search space reduces the values of

the parameters in the complexity bounds, but still leaves an exponential problem.

- Using premature termination of search based on a threshold on a "good" interpretation reduces the expected search. In particular, if the scene clutter is small enough relative to the noise in the data, the expected search becomes polynomial, otherwise it is a low order exponential.

To support the use of Hough transforms and premature termination of search, Eric Grimson and Daniel Huttenlocher have executed a formal analysis of these methods [28]. They have derived formal characterizations for the probability of false positives in the Hough space, as a function of the noise in the data and the characteristics of the Hough transform. These results provide a means of evaluating the efficacy of the Hough transform, and suggest that one should not, in general, rely on the Hough transform to fully solve the recognition problem, but rather that one should use it as a preprocessor, selecting out small subspaces within which the RAF method can be applied effectively. The results support the empirical observations concerning the reduction in search.

Grimson and Huttenlocher have also developed a formal characterization of thresholds for terminating search, relating analytic bounds on such thresholds to expected probabilities of errors. These formal results have been shown to agree with empirical evidence from several recognition systems.

Much of our earlier work with the RAF recognition system dealt with robotics environments and the recognition of industrial parts. We have continued this effort by integrating RAF into the HANDEY task-level planning system of Lozano-Pérez. We have also continued a pilot study of applying the technique to a very different domain, underwater localization. Specifically, we have considered the problem of determining the location of an autonomous underwater vehicle by matching sensory data obtained by the vehicle against bathymetric or other maps of the environment. Sensor modalities include active methods such as sonar, and passive methods such as pressure readings and doppler data from passing ships. We have conducted some early simulation experiments using RAF, together with strategies for acquiring sensory data to solve this localization problem, with excellent results.

Our formal analysis and our empirical experience both argue that the RAF approach to recognition fails to adequately deal with the issue of segmentation of the data into subsets that are likely to have come from a single object. While the Hough transform can help reduce this problem, it is model driven, and hence potentially very expensive when applied to large libraries of objects. As an alternative to this, David Jacobs has directly addressed the issue of generic grouping in an image [38]. Jacobs has derived r s for determining the probability that a set of edge fragments in an image is likely to have come from a single object. These measures consider simple measurements such as the separation of groups of edges, and the relative alignment of groups of edges. The recognition system, since it does not directly consider the object model, may occasionally be incorrect. However, tests of the system on a variety of images of two-dimensional and three-

dimensional scenes shows a remarkable and dramatic reduction in the search required to recognize objects from a library, and also is quite effective at identifying groups of edges coming from a single object. The effect of this grouping mechanism is particularly apparent when applied to libraries of objects, since the parameters computed by the grouping scheme can be used to do effective indexing into a library.

We have also continued to investigate the use of parallel architectures, such as the Connection Machine, to obtain significant performance improvements. Todd Cass has completed the development and implementation of a parallel recognition scheme for two dimensional scenes, on which he reported in the 1988 Proceedings . The system uses a careful Hough transform method. followed by a sampling scheme in the parameter space to find instances of an object and its pose. Typical performance of the method involves the correct identification and localization of heavily occluded objects, in scenes in which a large number of other parts are present, in under five seconds, using a 16K processor configuration of the Connection Machine. More recent work – mentioned earlier – has focused on integrating this recognition method with data provided by the Vision Machine.

References

[1] N. Ayache and O. D. Faugeras. Hyper: A new approach for the recognition and positioning of two-dimensional objecs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(1):44–54, 1986.

[2] H. G. Barrow and J. M. Tenenbaum. Recovering intrinsic scene characteristics from images. In A. R. Hanson and E. M. Riseman, editors, *Computer Vision Systems*, pages 3–26. Academic Press, New York, 1978.

[3] M. Bertero, T. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76:869–889, 1988.

[4] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *J. Roy. Stat. Soc.*, B34:75–83, 1972.

[5] G. Blelloch. Scans as primitive parallel operations. In *Proc. Intl. Conf. on Parallel Processing*, pages 355–362, 1987.

[6] J. Bliss. Velocity tuned spatio-temporal interpolation and approximation in vision. Master's thesis, Massachusetts Institute of Technology, 1985.

[7] Robert C. Bolles, Patrice Horaud, and M.J. Hannah. 3dpo: A three-dimensional part orientation system. In *Proceedings IJCAI*, pages 1116–1120, 1983.

[8] T.M. Breuel. Adaptive model base indexing. A.I. Memo 1008, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[9] R.A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2:14–23, 1987.

[10] Heinrich H. Bülthoff, James J. Little, and Tomaso Poggio. Parallel motion algorithm explains barber pole and motion capture illusion without "tricks". *J. Opt. Soc. Am.*, 4:34, 1987.

[11] Heinrich H. Bülthoff and Hanspeter A. Mallot. Interaction of different modules in depth perception. In *Proceedings of the International Conference on Computer Vision*, pages 295–305, London, England, June 1987. IEEE, Washington, DC.

[12] John F. Canny. Finding lines and edges in images. Technical Report TM-720, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1983.

[13] T.A. Cass. Robust parallel computation of 2D model-based recognition. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1988. Science Applications International Corp.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Roy. Stat. Soc.*, B 39:1–38, 1977.

[15] M. Drumheller and T. Poggio. On parallel stereo. In *Proc. IEEE Conf. on Robotics and Automation*, Washington, DC. 1986. IEEE.

[16] S. Edelman and S. Ullman. Reading cursive script by computer. In *Proceedings of the 42nd SPIE Conference*, pages 179–182, 1989.

[17] M. Fahle and Tomaso Poggio. Visual hyperacuity: Spatiotemporal interpolation in human vision. *Proc. R. Soc. Lond. B*, 213:451–477, 1980.

[18] E. Gamble, D. Geiger, T. Poggio, and D. Weinshall. Labeling edges and the integration of low-level visual modules. *IEEE Trans. Systems, Man and Cybernetics*, 19, 1989.

[19] E. Gamble and T. Poggio. Visual integration and detection of discontinuities: The key role of intensity edges. A.I. Memo 970, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1987.

[20] D. Geiger and F. Girosi. Mean field theory for surface reconstruction. In *Proceedings of the DARPA Image Understanding Workshop*, pages 617–630, McLean, VA, 1989. Science Applications International Corp.

[21] D. Geman and S. Geman. Relaxation and annealing with constraints. Complex Systems Technical Report 35, Division of Applied Mathematics, Brown University, Providence, RI, 1987.

[22] S. Geman and C. Graffigne. Markov random field image models and their applications to computer vision. In *Proc. Intl. Congress of Mathematicians*, 1987.

[23] Stuart Geman and Don Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:721–741, 1984.

[24] F. Girosi and T. Poggio. Networks and the best approximation property. A.I. Memo 1164, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.

[25] F. Girosi and T. Poggio. Networks for learning: a view from the theory of approximation of functions. In *Proc. of the Genoa Summer School on neural networks and their applications.* Prentice Hall, 1989.

[26] F. Girosi and T. Poggio. Extensions of a theory of networks for approximation and learning: dimensionality reduction and clustering. A.I. Memo 1167, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, March 1990.

[27] W. Eric L. Grimson. *From Images to Surfaces.* MIT Press, Cambridge, Mass., 1981.

[28] W.E.L. Grimson and D.P. Huttenlocher. On the sensitivity of the hough transform for object recognition. A.I. Memo 1044, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA. 1988.

[29] W.E.L. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:469–482, 1987.

[30] D. Hillis. *The Connection Machine.* The MIT Press, Cambridge, MA, 1985.

[31] A. Hurlbert and T. Poggio. Spotlight on attention. *Trends in Neurosciences*, 8:309–311, 1985.

[32] A. Hurlbert and T. Poggio. Learning a color algorithm from examples. A.I. Memo 909, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[33] A. Hurlbert and T. Poggio. A network for image segmentation using color. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems - I*, pages 297–304. Morgan Kaufmann Publishers, 1989.

[34] A.C. Hurlbert. *The computation of color.* PhD thesis, Massachusetts Institute of Technology, 1989.

[35] Anya Hurlbert and Tomaso Poggio. Do computers need attention? *Nature*, 321(12), 1986.

[36] Anya Hurlbert and Tomaso Poggio. Synthetizing a color algorithm from examples. *Science*, 239:482–485, 1988.

[37] D.P. Huttenlocher and S. Ullman. Recognizing rigid objects by aligning them with an image. A.I. Memo 937, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge MA, 1987.

[38] D.W. Jacobs. The use of grouping in visual object recognition. A.I. Technical Report 1023, MIT, Cambridge, MA, January 1988.

[39] Michael Kass. Computing visual correspondence. In *From Pixels to Predicates*, pages 78–92. Ablex Publishing Corporation, Norwood, NH, 1986.

[40] Christof Koch, Jose Marroquin, and Alan Yuille. Analog 'neuronal' networks in early vision. A.I. Memo No. 751. Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1985.

[41] H. Lee. Estimating the illuminant color from the shading of a smooth surface. A.I. Memo 1068, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA. August 1988.

[42] W. Lim. Fast algorithms for labelling connected components in 2-D arrays. TMC Technical Report NA86-1, Thinking Machines Corporation, Cambridge, MA, December 1986.

[43] J. Little, G. Blelloch, and T. Cass. How to program the connection machine for computer vision. In *Proc. Workshop on Comp. Architecture for Pattern Analysis and Machine Intell.*, 1987.

[44] J. Little. G. Blelloch, and T. Cass. Parallel algorithms for computer vision on the connection machine. In *Proceedings of the DARPA Image Understanding Workshop*, pages 628–638, McLean, VA, 1987. Science Applications International Corp.

[45] J. Little, H. Buelthoff, and T. Poggio. Parallel optical flow computation. In *Proceedings of the DARPA Image Understanding Workshop*, pages 915–920, McLean, VA, 1987. Science Applications International Corp.

[46] J. Little, H. Buelthoff, and T. Poggio. Parallel optical flow using local voting. In *Proceedings of the International Conference on Computer Vision*, Washington, DC, 1988. IEEE.

[47] David G. Lowe. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, Boston, 1986.

[48] James V. Mahoney. Image chunking: Defining spatial building blocks for scene analysis. Master's thesis, Massachusetts Institute of Technology, 1986.

[49] David Marr and Tomaso Poggio. Cooperative computation of stereo disparity. *Science*, 194:283–287, 1976.

[50] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *J. Amer. Statistical Assoc.*, 82:76–89, 1987.

[51] J.L. Marroquin. Deterministic Bayesian estimation of Markovian random fields with applications to computational vision. In *Proceedings of the International Conference on Computer Vision*, Washington, DC, 1987. IEEE.

[52] Jose L. Marroquin. *Probabilistic Solution of Inverse Problems.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1985.

[53] H.K. Nishihara. Practical real-time imaging stereo matcher. *Optical Engineering*, 23(5):536–545, 1984.

[54] H.K. Nishihara and P.A. Crossley. Measuring photolithographic overlay accuracy and critical dimensions by correlating binarized laplacian of gaussian convolutions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-10(1):4–16, 1988.

[55] T. Poggio and F. Girosi. A theory of networks for approximation and learning. A.I. Memo 1140. Artificial Intelligence Laboratory. Massachusetts Institute of Technology, Cambridge. MA, July 1989.

[56] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, 247:978–982, 1990.

[57] T. Poggio and the Staff of the A.I. Laboratory. Progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1985. Science Applications International Corp.

[58] T. Poggio and the Staff of the A.I. Laboratory. Progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1988. Science Applications International Corp.

[59] Tomaso Poggio, Edward B. Gamble, and James J. Little. Parallel integration of vision modules. *Science*, 242:436–440 and cover, 1988.

[60] Tomaso Poggio, J. Little, E. Gamble, W. Gillett, D. Geiger, D. Weinshall, M. Villalba, N. Larson, T. Cass, H. Bülthoff, M. Drumheller, P. Oppenheimer, W. Yang, and A. Hurlbert. The MIT Vision Machine. In *Proceedings of the DARPA Image Understanding Workshop*, Cambridge, MA, April 1988. Morgan Kaufmann, San Mateo, CA.

[61] Tomaso Poggio, K.R.K. Nielsen, and H.Kieth Nishihara. Zero-crossings and spatiotemporal interpolation in vision:aliasing and electrical coupling between sensors. A.I. Memo No. 675, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1982.

[62] Tomaso Poggio and Werner Reichardt. Visual control of orientation behaviour in the fly: Part II: Towards the underlying neural interactions. *Quart. Rev. Biophysics*, 9:377–438, 1976.

[63] T.D. Sanger. Stereo disparity computation using gabor filters. *Biological Cybernetics*, 59:405–418, 1988.

[64] T.D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–473, 1989.

[65] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems.* W.H.Winston, Washington, D.C., 1977.

[66] V. Torre and T. Poggio. On edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 8:147–163, 1986.

[67] Lewis W. Tucker, Carl R. Feynman, and Donna M. Fritzsche. Object recognition using the Connection Machine. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, Ann Arbor, MI, June 1988.

[68] Shimon Ullman. Visual routines. *Cognition*, 18, 1984.

[69] Shimon Ullman. An approach to object recognition: Aligning pictorial descriptions. A.I. Memo No. 931, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA. December 1986.

[70] A. Verri, F. Girosi, and V. Torre. The mathematical properties of the 2D motion field: from singular points to motion parameters. *J. Opt. Soc. Am. A*, 6:698–712, 1989.

[71] A. Verri and T. Poggio. Motion field and optical flow: Qualitative properties. A.I. Memo 917, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1986.

[72] H. Voorhees. Finding texture boundaries in images. Master's thesis, Massachusetts Institute of Technology, June 1987.

[73] H. Voorhees and T. Poggio. Detecting blobs as textons in natural images. In *Proceedings of the DARPA Image Understanding Workshop*, pages 892–899, McLean, VA, 1987. Science Applications International Corp.

[74] H. Voorhees and T. Poggio. Computing texture boundaries from images. *Nature*, 333:364–367, 1988.

[75] Harry L. Voorhees. Finding texture boundaries in images. Master's thesis, Massachusetts Institute of Technology, 1987.

[76] Allen M. Waxman. Image flow theory: A framework for 3-D inference from time-varying imagery. In C. Brown, editor, *Advances in Computer Vision.* Erlbaum, New Jersey, 1987.

[77] Alan L. Yuille and Tomaso Poggio. A generalized ordering constraint for stereo correspondence. A.I. Memo No. 777, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1984.

Bibliography of Works done at the AI Lab under this Contract

References

[1] R. Basri and S. Ullman. The alignment of objects with smooth surfaces. In *Proceedings of the International Conference on Computer Vision*, pages 482–488, Washington, DC, 1988. IEEE.

[2] M. Bertero, T. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76:869–889, 1988.

[3] G. Blelloch. Scans as primitive parallel operations. In *Proc. Intl. Conf. on Parallel Processing*, pages 355–362, 1987.

[4] G. Blelloch and J. Little. Parallel solutions to geometric problems on the scan model of computation. A.I. Memo 952, Artificial Intelligence Laboratory, Massachusetts Institute of Technology. Cambridge, MA, 1987.

[5] G. Blelloch and C. Rosenberg. Network learning on the connection machine. In *Proc. Intl. Joint. Conf. on Artificial Intell.*, pages 323–326, 1987.

[6] J. Bliss. Velocity tuned spatio-temporal interpolation and approximation in vision. Master's thesis, Massachusetts Institute of Technology, 1985.

[7] David J. Braunegg. An alternative to using the 3-D delaunay tessellation for representing freespace. A.I. Memo 1185, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1989.

[8] David J. Braunegg. Location recognition using stereo vision. A.I. Memo 1186, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.

[9] David J. Braunegg. Stereo feature matching in disparity space. A.I. Memo 1184, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1989.

[10] David J. Braunegg. *MARVEL: A System for Recognizing World Locations with Stereo Vision.* PhD thesis, Massachusetts Institute of Technology, June 1990.

[11] David J. Braunegg. Stereo feature matching in disparity space. In *Proc. IEEE International Conference on Robotics and Automation*, Cincinnati, OH, May 1990.

[12] T.M. Breuel. Adaptive model base indexing. A.I. Memo 1008, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[13] R. Brooks. Intelligence without representation. In *Proc. Foundations of AI Workshop*, Cambridge, MA. 1987. The MIT Press.

[14] R. Brooks and J. Connell. Asynchronous distributed control systems for a mobile robot. In *Proc. S.P.I.E.*, page 727. 1986.

[15] R. Brooks, A. Flynn, and T. Marill. Self calibration of motion and stereo vision for mobile robot navigation. A.I. Memo 984, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[16] R.A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*. RA-2:14–23. 1987.

[17] H. Buelthoff, J. Little, and T. Poggio. Parallel motion algorithm explains barber pole and motion capture illusion without "tricks". *J. Opt. Soc. Am.*, 4:34, 1987.

[18] H. Buelthoff, J. Little, and T. Poggio. A parallel algorithm for real-time computation of motion. *Nature*. 337:549–553, 1989.

[19] H. Buelthoff, J. Little, and T. Poggio. A parallel motion algorithm consistent with psychophysics and physiology. In *Proc. of the IEEE Workshop on Visual Motion*, pages 165–172, Washington, DC, 1989. IEEE.

[20] H. Buelthoff and H.A. Mallot. Interaction of different modules in depth perception. In *Proceedings of the International Conference on Computer Vision*, pages 295–305, 1987.

[21] H Buelthoff and H.A. Mallot. Interaction of depth modules: stereo and shading. *Journal of the Optical Society of America*, 5:1749–1758, 1988.

[22] Heinrich Buelthoff and H.A. Mallot. Interaction of different modules in depth perception. A.I. Memo 965, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[23] J.F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[24] T.A. Cass. Parallel computation in model-based recognition. Master's thesis, Massachusetts Institute of Technology. June 1988.

[25] T.A. Cass. A robust implementation of 2D model-based recognition. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, 1988.

[26] T.A. Cass. Robust parallel computation of 2D model-based recognition. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1988. Science Applications International Corp.

[27] D.T. Clemens. The recognition of two-dimensional modeled objects in images. Master's thesis, Massachusetts Institute of Technology, 1986.

[28] J. Connell. Task oriented spatial representation for distributed systems. A.I. Memo 823, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1986.

[29] M. Drumheller. Connection machine stereo matching. In *Proceedings AAAI*, pages 748–753, 1986.

[30] M. Drumheller and T. Poggio. On parallel stereo. In *Proc. IEEE Conf. on Robotics and Automation*, Washington, DC, 1986. IEEE.

[31] S. Edelman. Reading cursive handwriting. *Perception*, 18:524, 1989.

[32] S. Edelman, H. Buelthoff, and D. Weinshall. Exploring representation of 3D objects for visual recognition. *Invest. Ophthalm. Vis. Science*, 30, 1989.

[33] S. Edelman, H. Buelthoff, and D. Weinshall. Integrating information for visual recognition of 3D objects. *Perception*, 18:517, 1989.

[34] S. Edelman, H. Buelthoff, and D. Weinshall. Stimulus familiarity determines recognition strategy for novel 3D objects. A.I. Memo 1138, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[35] S. Edelman and T. Poggio. Integrating visual cues for object segmentation and recognition. *Optics News*, 15:8–15, 1989.

[36] S. Edelman and T. Poggio. Representations in high-level vision: Reassessing the inverse optics paradigm. In *Proceedings of the DARPA Image Understanding Workshop*, pages 94–949, McLean, VA, 1989. Science Applications International Corp.

[37] S. Edelman and S. Ullman. Reading cursive script by computer. In *Proceedings of the 42nd SPIE Conference*, pages 179–182, 1989.

[38] S. Edelman and D. Weinshall. Computational vision: a critical review. A.I. Memo 1158, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.

[39] S. Edelman and D. Weinshall. A self-organizing multiple-view representation of 3D objects. A.I. Memo 1146, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, August 1989.

[40] S. Edelman. D. Weinshall, H. Buelthoff, and T. Poggio. A model of the acquisition of object representations in human 3D visual recognition. In *Proceedings of the NATO Advanced Research Workshop on Robots and Biological Systems*. Berlin, 1989. Springer-Verlag.

[41] E. Gamble, D. Geiger, T. Poggio, and D. Weinshall. Labeling edges and the integration of low-level visual modules. *IEEE Trans. Systems, Man and Cybernetics*, 19, 1989.

[42] E. Gamble and T. Poggio. Visual integration and detection of discontinuities: The key role of intensity edges. A.I. Memo 970, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1987.

[43] E.B. Gamble. A comparison of hardware implementations for low-level vision algorithms. A.I. Memo No. 1173, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, November 1989.

[44] D. Geiger and F. Girosi. Mean field theory for surface reconstruction. In *Proceedings of the DARPA Image Understanding Workshop*, pages 617–630, McLean, VA, 1989. Science Applications International Corp.

[45] D. Geiger and F. Girosi. Neural networks from coupled markov random fields via mean field theory. In *Proc. IEEE Conference on Neural Information Processing Systems - Natural and Synthetic*, 1989.

[46] D. Geiger and F. Girosi. Parallel and deterministic algorithms for MRFs: surface reconstruction and integration. A.I. Memo 1114, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, May 1989.

[47] D. Geiger and T. Poggio. Level crossings and the panum area. In *Proc. IEEE Computer Society Workshop on Computer Vision*, pages 211–214, Washington, DC, 1987. IEEE.

[48] D. Geiger and T. Poggio. An optimal scale for edge detection. In *Proceedings IJCAI*, pages 645–748. 1987.

[49] D. Geiger and T. Poggio. An optimal scale for edge detection. A.I. Memo 1078, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1988.

[50] W. Gillett. Issues in parallel stereo matching. Master's thesis, Massachusetts Institute of Technology, 1988.

[51] F. Girosi and T. Poggio. Networks and the best approximation property. A.I. Memo 1164, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, October 1989.

[52] F. Girosi and T. Poggio. Networks for learning: a view from the theory of approximation of functions. In *Proc. of the Genoa Summer School on neural networks and their applications.* Prentice Hall, 1989.

[53] F. Girosi and T. Poggio. Representation properties of networks: Kolmogorov's theorem is irrelevant. *Neural Computation*, 1:465–469, 1989.

[54] F. Girosi and T. Poggio. Extensions of a theory of networks for approximation and learning: dimensionality reduction and clustering. A.I. Memo 1167, Artificial Intelligence Laboratory, Massachusetts Institute of Technology. Cambridge, MA, March 1990.

[55] F. Girosi, A. Verri, and V. Torre. Constraints on the computation of optical flow. In *IEEE Workshop on Visual Motion*, pages 116–124, 1989.

[56] W.E.L. Grimson. Combinatorics of object recognition in cluttered environments using constrained search. A.I. Memo 1091, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1988.

[57] W.E.L Grimson. The combinatorics of object recognition in cluttered environments using constrained search. In *Proceedings of the International Conference on Computer Vision*, 1988.

[58] W.E.L. Grimson. Determining object pose for grasping and manipulation. In M. Goodale, editor, *Vision and Action: The Control of Grasping.* Ablex Publishing Corporation, 1988.

[59] W.E.L. Grimson. On the recognition of parameterized 2D objects. A.I. Memo 985, Artificial Intelligence Laboratory. Massachusetts Institute of Technology, Cambridge, MA, 1988.

[60] W.E.L. Grimson. On the recognition of curved objects in two dimensions. A.I. Memo 983, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[61] W.E.L. Grimson and D.P. Huttenlocher. On the sensitivity of the hough transform for object recognition. In *Proceedings of the International Conference on Computer Vision*, 1988.

[62] W.E.L. Grimson and D.P. Huttenlocher. On the sensitivity of the hough transform for object recognition. A.I. Memo 1044, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1988.

[63] W.E.L. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:469–482, 1987.

[64] W.E.L. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. A.I. Memo 841, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[65] J. Heel. Dynamical systems and motion vision. A.I. Memo 1037, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, April 1988.

[66] E.C. Hildreth and S. Ullman. The computational study of vision. A.I. Memo 1038, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1988.

[67] D. Hillis. *The Connection Machine.* The MIT Press, Cambridge, MA, 1985.

[68] B.K.P. Horn. *Robot Vision.* The MIT Press, Cambridge, MA, 1986.

[69] B.K.P. Horn and M. Brooks. *Seeing Shape from Shading.* The MIT Press, Cambridge, MA, 1989.

[70] I. Horswill and R. Brooks. Situated vision in a dynamic world: Chasing objects. In *Proceedings AAAI*, pages 796–800, 1988.

[71] A. Hurlbert, H.C. Lee, and H. Buelthoff. Cues to the color of the illuminant. *Invest. Ophthalm. Vis. Science Suppl.*, 30:221, 1989.

[72] A. Hurlbert and T. Poggio. Spotlight on attention. *Trends in Neurosciences*, 8:309–311, 1985.

[73] A. Hurlbert and T. Poggio. Spotlight on attention. A.I. Memo 817, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1985.

[74] A. Hurlbert and T. Poggio. Do computers need attention? *Nature*, 321, 1986.

[75] A. Hurlbert and T. Poggio. Visual attention in brains and computers. A.I. Memo 915, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1986.

[76] A. Hurlbert and T. Poggio. Learning a color algorithm from examples. A.I. Memo 909, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[77] A. Hurlbert and T. Poggio. Making machines (and AI) see. *Daedalus*, 117:213–239, 1988.

[78] A. Hurlbert and T. Poggio. Synthesizing a color algorithm from examples. *Science*, 27:116–120, 1988.

[79] A. Hurlbert and T. Poggio. A network for image segmentation using color. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems - I*, pages 297–304. Morgan Kaufmann Publishers, 1989.

[80] A.C. Hurlbert. *The computation of color.* PhD thesis, Massachusetts Institute of Technology, 1989.

[81] A.C. Hurlbert and T. Poggio. Learning a color algorithm from examples. In *Neural Information Processing Systems: Proceedings of the Neural Information Processing Conference*, pages 622–631, New York, NY, 1988. American Institute of Physics.

[82] D.P. Huttenlocher and S. Ullman. Object recognition using alignment. In *Proceedings of the International Conference on Computer Vision*, pages 102–111, 1987.

[83] D.P. Huttenlocher and S. Ullman. Recognizing rigid objects by aligning them with an image. A.I. Memo 937, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge MA, 1987.

[84] D.W. Jacobs. The use of grouping in visual object recognition. A.I. Technical Report 1023, MIT, Cambridge, MA, January 1988.

[85] H. Lee. Estimating the illuminant color from the shading of a smooth surface. A.I. Memo 1068, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, August 1988.

[86] W. Lim. Fast algorithms for labelling connected components in 2-D arrays. TMC Technical Report NA86-1, Thinking Machines Corporation, Cambridge, MA, December 1986.

[87] W. Lim. Using occluding contours for object recogition. In *Proceedings of the DARPA Image Understanding Workshop*, pages 909–914, 1987.

[88] W. Lim. *Shape Recognition in the Rocks World.* PhD thesis, Massachusetts Institute of Technology, May 1988.

[89] J. Little. Parallel algorithms for computer vision on the connection machine. A.I. Memo 928, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1986.

[90] J. Little. The structure of vision modules on a fine-grained machine. In *Proceedings of SPIE Conf. on Advances in Intelligent Robotics Systems*, Bellingham, WA, 1987. SPIE.

[91] J. Little, G. Blelloch, and T. Cass. How to program the connection machine for computer vision. In *Proc. Workshop on Comp. Architecture for Pattern Analysis and Machine Intell.*, 1987.

[92] J. Little, G. Blelloch, and T. Cass. Parallel algorithms for computer vision on the connection machine. In *Proceedings of the International Conference on Computer Vision*, pages 587–591, 1987.

[93] J. Little. G. Blelloch, and T. Cass. Parallel algorithms for computer vision on the connection machine. In *Proceedings of the DARPA Image Understanding Workshop*, pages 628–638, McLean, VA, 1987. Science Applications International Corp.

[94] J. Little and H. Buelthoff. Parallel computation of optical flow. A.I. Memo 929, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[95] J. Little, H. Buelthoff, and T. Poggio. Parallel optical flow computation. In *Proceedings of the DARPA Image Understanding Workshop*, pages 915–920. McLean, VA, 1987. Science Applications International Corp.

[96] J. Little, H. Buelthoff, and T. Poggio. Parallel optical flow using local voting. In *Proceedings of the International Conference on Computer Vision*, Washington, DC, 1988. IEEE.

[97] J. Little and T. Poggio. The vision machine project: Integrating early vision modules. In *Proceedings 1988 Spring Symposium Series - Physical and Biological Approaches to Computational Vision*, pages 85–87. AAAI, 1988.

[98] J. Little, T. Poggio, and E.B. Gamble. Seeing in parallell: The vision machine. *Intl. J. of Supercomputer Applications*, 2:13–28, 1988.

[99] J. Little and A. Verri. Analysis of differential and matching techniques for optical flow. A.I. Memo 1066, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1988.

[100] J. Mahoney. Image chunking: Defining spatial building blocks for scene analysis. Master's thesis, Massachusetts Institute of Technology, 1986.

[101] J. Mahoney. Image chunking: Defining spatial building blocks for scene analysis. A.I. Laboratory Technical Report 980, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, January 1987.

[102] H.A. Mallot, H. Buelthoff, and J. Little. Neural architecture for optical flow computation. A.I. Memo 1067, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, March 1989.

[103] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *J. Amer. Statistical Assoc.*, 82:76–89, 1987.

[104] J. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. A.I. Memo 897, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, March 1987.

[105] J.L. Marroquin. Deterministic Bayesian estimation of Markovian random fields with applications to computational vision. In *Proceedings of the International Conference on Computer Vision*. Washington, DC, 1987. IEEE.

[106] B. Moore and T. Poggio. Representations properties of multilayer feedforward networks. In *Abstracts of the First Annual INNS Meeting*, page 502, New York, 1988. Pergamon Press.

[107] T. Poggio. Computer vision. In E. Clementi and S. Chin, editors, *Biological and Artificial Intelligence Systems*. pages 471–483. ESCOM Science Publishers, Leiden, The Netherlands, 1988.

[108] T. Poggio. A parallel vision machine that learns. In R. Cotterill, editor, *Models of Brain Function*, pages 51–88. University of Cambridge Press, Cambridge, UK, 1989.

[109] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.

[110] T. Poggio, E. Gamble, and J. Little. Parallel integration of vision modules. In *Proceedings 1988 Spring Symposium Series - Physical and Biological Approaches to Computational Vision*, pages 88–95. AAAI, 1988.

[111] T. Poggio, E. Gamble, and J. Little. Parallel integration of vision modules. *Science*, 242:436–440 and cover, 1988.

[112] T. Poggio and F. Girosi. A theory of networks for approximation and learning. A.I. Memo 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, July 1989.

[113] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*, 247:978–982, 1990.

[114] T. Poggio and C. Koch. Ill-posed problems in early vision: from computational theory to analog networks. *Proc. R. Soc. Lond. B*, 226:303–323, 1985.

[115] T. Poggio, J. Little, E. Gamble, W. Gillett. D. Geiger, D. Weinshall, M. Villalba, N. Larson, T. Cass, H. Buelthoff, M. Drumheller, P. Oppenheimer, W. Yang, and A. Hurlbert. The vision machine. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1988. Science Applications International Corp.

[116] T. Poggio and the Staff of the A.I. Laboratory. Progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1985. Science Applications International Corp.

[117] T. Poggio and the Staff of the A.I. Laboratory. Progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1987. Science Applications International Corp.

[118] T. Poggio and the Staff of the A.I. Laboratory. Progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1988. Science Applications International Corp.

[119] T. Poggio and the Staff of the A.I. Laboratory. Progress in understanding images. In *Proceedings of the DARPA Image Understanding Workshop*, McLean, VA, 1989. Science Applications International Corp.

[120] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. *Nature*, 317:314–319, 1985.

[121] T. Poggio, V. Torre, and C. Koch. Computational vision and regularization theory. In M.A. Fischler and O. Firschein, editors, *Readings in Computer Vision*. Morgan Kaufmann Publishers, Los Altos, CA, 1987.

[122] T. Poggio and A. Verri. Regularization theory and shape constraint. A.I. Memo 916, Artificial Intelligence Laboratory, Massachusetts Institute of Technology. Cambridge, MA. 1986.

[123] T. Poggio, H. Voorhees, and A. Yuille. A regularized solution to edge detection. *Journal of Complexity*, 4:106–123, 1988.

[124] T. Poggio, W. Yang, and V. Torre. Optical flow: Computational properties and networks, biological and analog. In R. Durbin, C. Miall, and G. Mitchison, editors, *The Computing Neuron*, pages 355–370. Addision-Wesley, Reading, MA, 1988.

[125] T.D. Sanger. Optimal unsupervised learning. *Neural Networks*, 1:127, 1988.

[126] T.D. Sanger. Stereo disparity computation using gabor filters. *Biological Cybernetics*, 59:405–418, 1988.

[127] T.D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–473, 1989.

[128] T.D. Sanger. Optimal unsupervised learning in feedforward neural networks. A.I. Laboratory Technical Report 1086, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[129] T.D. Sanger. An optimality principle for unsupervised learning. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems - I*, pages 11–19. Morgan Kaufmann, San Mateo, CA, 1989.

[130] E. Saund. Dimensionality-reduction using connectionist networks. A.I. Memo 941, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1987.

[131] A. Sha'ashua and S. Ullman. Structural saliency: the detection of globally salient structures using a locally connected network. In *Proceedings of the International Conference on Computer Vision*, pages 321–327, Washington, DC, 1988. IEEE.

[132] V. Torre and T. Poggio. On edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:147–163. 1986.

[133] S. Ullman. An approach to object recognition: Aligning pictorial descriptions. A.I. Memo 931. Artificial Intelligence Laboratory. Massachusetts Institute of Technology, Cambridge, MA, December 1986.

[134] S. Ullman. Aligning pictorial descriptions: an approach to object recognition. *Cognition*, 32:193–254, 1989.

[135] S. Ullman and R. Basri. Recognition by linear combinations of models. A.I. Memo No. 1152, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[136] S. Ullman and A. Sha'ashua. Structural saliency: The detection of globally salient structures using a locally connected network. A.I. Memo 1061, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, July 1988.

[137] A. Verri, F. Girosi, and V. Torre. The mathematical properties of the 2D motion field: from singular points to motion parameters. *J. Opt. Soc. Am. A*, 6:698–712, 1989.

[138] A. Verri, F. Girosi, and V. Torre. The mathematical properties of the 2D motion field: from singular points to motion parameters. In *IEEE Workshop on Visual Motion*, pages 190–200, 1989.

[139] A. Verri and T. Poggio. Motion field and optical flow: Qualitative properties. A.I. Memo 917, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 1986.

[140] A. Verri and T. Poggio. Against quantitative optical flow. In *Proceedings of the International Conference on Computer Vision*, pages 171–180, 1987.

[141] A. Verri and T. Poggio. Qualitative information in the optical flow. In *Proceedings of the DARPA Image Understanding Workshop*, pages 825–834, McLean, VA, 1987. Science Applications International Corp.

[142] A. Verri and T. Poggio. Motion field and optical flow: qualitative properties. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:490–498, 1989.

[143] H. Voorhees. Finding texture boundaries in images. Master's thesis, Massachusetts Institute of Technology, June 1987.

[144] H. Voorhees. Finding texture boundaries in images. A.I. Laboratory Technical Report 968, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, June 1987.

[145] H. Voorhees and T. Poggio. Detecting blobs as textons in natural images. In *Proceedings of the DARPA Image Understanding Workshop*, pages 892–899, McLean, VA. 1987. Science Applications International Corp.

[146] H. Voorhees and T. Poggio. Detecting textons and texture boundaries in natural images. In *Proceedings of the International Conference on Computer Vision*, pages 250–258, 1987.

[147] H. Voorhees and T. Poggio. Computing texture boundaries from images. *Nature*, 333:364–367, 1988.

[148] D. Weinshall. Qualitative depth and shape from stereo in agreement with psychophysical evidence. A.I. Memo 1007, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, December 1987.

[149] D. Weinshall. Application of qualitative depth and shape from stereo. In *Proceedings of the International Conference on Computer Vision*, pages 144–148, Washington, DC, 1988. IEEE.

[150] D. Weinshall. Seeing 'ghost' solutions in stereo vision. A.I. Memo 1073, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, September 1988.

[151] D. Weinshall. Direct computation of 3D shape and motion invariants. A.I. Memo 1131, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, May 1989.

[152] D. Weinshall, S. Edelman, and H. Buelthoff. A self-organizing multiple-view representation of 3D objects. In D. Touretzky, editor, *Natural Information Processing Systems - II.* Morgan Kaufmann, San Mateo, CA, 1990.

[153] W. Yang. A charge-coupled device architecture for on focal plane image signal processing. In *Proceedings of International Symposium on VLSI Technology, Systems, and Applications*, pages 266–270, 1989.

[154] W. Yang and A. Chiang. VLSI processor architectures for computer vision. In *Proceedings of the DARPA Image Understanding Workshop*, pages 193–199, McLean, VA, 1989. Science Applications International Corp.

[155] W. Yang and A. Chiang. A full-fill factor CCD imager with integrated signal processors. *Technical Digest of International Solid State Circuits Conference*, pages 218–219, 1990.

[156] A. Yuille and T. Poggio. Scaling theorems for zero-crossings. In W. Richards and S. Ullman, editors, *Image Understanding 1985*. Ablex Publishing, 1985.

[157] A. Yuille and T. Poggio. Scaling theorems for zero-crossings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:15–25, 1986.

[158] A. Yuille and T. Poggio. Scaling and fingerprint theorems for zero-crossings. In C. Brown, editor, *Advances in Computer Vision*. pages 47–78. Lawrence Erlbaum Assocs., Hillsdale. NJ, 1988.